Commodore
COMPUTER
CLUB
32
L. 3.500
La rivista degli utenti di sistemi Commodore
25 giugno 1986 - Anno V - N. 32 - Sped. Abb. Post. GR. III/70 - CR - Distr: MePe
Mundial: cinque sport
da giocare col 64
Grafica: la gestione
dei 4 banchi di memoria
Inserto: gli errori
nelle stringhe
Disegnare
col computer
Antidecompilatore
per C/64
systems

Sommario

INSERTO

GLI ERRORI DOVUTI ALLA MANIPOLAZIONE DI STRINGHE

RUBRICHE

Direttore: Alessandro de Simone
Redazione/collaboratori: Claudio Baiocchi, Carlo e Lorenzo Barazzetta, Giovanni Bellù, Simone Bettola, Andrea e Alberto Boriani, Diego e Federico Canetta, Giancarlo Castagna, Umberto Colapicchioni, Pasquale D'Andreti, Maurizio Dell'Abate, Valerio Ferri, Luca Galluzzi, Michele Maggi, Giancarlo Mariani, Marco Miotti, Flavio Molinari, Claudio Mueller, Massimo Pollutri, Carla Rampi, Fabio Sorgato, Giovanni Verrelli, Antonio Visconti.
Segreteria di redazione: Maura Ceccaroli, Piera Perin
Ufficio Grafico: Mary Benvenuto, Arturo Ciaglia
Direzione, redazione, pubblicità: V.le Famagosta, 75 - 20142 Milano - Tel. 02/8467348
Pubblicità: Milano: Leandro Nencioni (direttore vendite), Giorgio Ruffoni, Roberto Sghirinzetti Claudio Tidone - V.le Famagosta, 75 - 20142 Milano - Tel. 02/8467348
- Emilia Romagna: Spazio E - P.zza Roosevelt, 4 - 40123 Bologna - Tel. 051/236979
- Toscana, Marche, Umbria: Mercurio Srl - via Rodari, 9 - San Giovanni Valdarno (Ar) - Tel. 055/947444
- Lazio, Campania: Spazio Nuovo - via P. Foscari 70 - 00139 Roma - Tel. 06/8109679

Segretaria: Lilliana Degiorgi - **Abbonamenti:** Marina Vantini
Tariffe: prezzo per copia L. 3.500. Abbonamento annuo (11 fascicoli) L. 35.000. Estero: il doppio.
Abbonamento cumulativo alle riviste Computer e Commodore Computer Club L. 70.000
I versamenti vanno indirizzati a: Systems Editoriale Srl mediante assegno bancario o utilizzando il c/c postale n. 37952207
Composizioni: Systems Editoriale Srl - **Fotolito:** Systems Editoriale Srl
Stampa: La Litografica S.r.l. - Busto Arsizio (VA)
Registrazione: Tribunale di Milano n. 370 del 2/10/82 - Direttore Responsabile: Michele Di Pisa
Sped. in abb. post. gr. III - Pubblicità inferiore al 70% - **Distrib:** MePe, via G. Carcano 32 - Milano

Sport e computer

Come arrampicarsi sugli specchi per giustificare un legame tra due attività in contrasto tra loro

di Alessandro de Simone

Nonostante la provocatoria introduzione parleremo (brevemente) a ruota libera, stavolta, sulle possibili connessioni tra attività sportive e calcolatori.

Non ci riferiamo certo allo sport inteso come quell'attività che, comodamente appollaiati in poltrona, ci consente di esaminare al rallenty lo spettacolare goal segnato dal nostro beniamino; nè ci sfiora per la testa il desiderio di parlare dei numerosi programmi che aiutano la gente a far tredici (e a diminuire nel contempo le possibilità di grosse vincite).

Più interessante è di certo l'utilizzo del computer (in senso lato) per controllare l'attività sportiva di ciascuno di noi senza esagerare negli sforzi nè, d'altra parte, indurre in pigrizie sedentarie. Programmi di diete (che quest'anno, come i precedenti, ci siamo rifiutati di pubblicare a causa della loro inflazione) ci fanno solo sospirare di fronte alla nostra impotenza a contenere il bisogno calorico o a dimiuirlo con digiuni pannelliani.

Alcuni dubbi

rimangono sulla validità dei giochi che trasformano il calcola tore in un palcoscenico in cui simpatici burrattini, da noi manovrati, si agitano, sperano, soffrono, esultano o si avviliscono a seconda di vittorie clamorose o cocenti sconfitte.

Sono sempre stato contrario ai video giochi ma sono costretto ad ammettere che alcuni di essi, soprattutto recenti, sono vere e proprie simulazioni degne di ammirazione.

E se, in procinto di assistere alle imprese dell'Italia al Mundial, giocheremo al noto "Soccer" per ingannare l'attesa, non faremo certo male a nessuno sognando di essere al centro del campo mentre milioni di telespettatori seguono le nostre mosse col fiato sospeso...

Amiga

☐ **In America è stato presentato il nuovo computer della Commodore (Amiga). Di che cosa si tratta?**
(Massimo Ferrato, Domodossola)

• La "novità" dell'Amiga non è più... tale: da molto tempo, infatti, (cioè da circa un anno) è commercializzata negli USA ed anzi rappresenta il nuovo cavallo di battaglia della Commodore che intende così entrare nella competitivo mercato del Professional Computer.

Il nuovo calcolatore si rivolge, infatti, agli affezionati del C/64 che, consci dei limiti tecnici del piccolo computer, desiderano passare a calcolatori più potenti senza però abbandonare le caratteristiche grafiche, sonore e, più in generale, ludiche e di svago che ha sempre caratterizzato i prodotti Commodore.

All'interno del nuovo computer è infatti presente, tra gli altri, il potente microprocessore 68000 a sedici bit che non ha nulla da invidiare, quanto a velocità e potenza, all'8086 montati sui più diffusi MS/DOS. Sembra, inoltre, che possa diventare compatibile con quest'ultimo standard e, se questa informazione corrisponde al vero, non farebbe rimpiangere il mancato acquisto di un computer IBM o di altri che adottino il sistema operativo che va per la maggiore.

La grafica a colori in altissima risoluzione (sembra di osservare diapositive invece che semplici disegni), le capacità di sonorizzazione (dispone di un'uscita stereo), la velocità di esecuzione, la possibilità di connettersi direttamente ad "aggeggi" compatibili con lo standard 68000, sembrano porre il nuovo prodotto in una posizione invidiabile.

Il prezzo, per finire lietamente, pur se decisamente maggiore di un C/64 non è comunque elevato soprattutto se si confrontano le prestazioni.

Come è nostra abitudine, onde evitare facili entusiasmi, aspetteremo che in Italia venga definita con molta chiarezza la politica commerciale che i dirigenti della Commodore intendono adottare per il rivoluzionario prodotto. Ciò significa, in altre parole, che non siamo intenzionati a funger da cassa di risonanza per parlare di prodotti che non vengono, in seguito, adeguatamente sostenuti per ciò che riguarda programmi, accessori, notizie ed altro.
Il Plus/4 ci ha insegnato qualcosa, e ne faremo tesoro.

Tavolette grafiche

☐ **Che cosa è una tavoletta grafica? Esistono simili accessori per il mio Plus/4?**
(Marco Bologna, Torino)

• La tavoletta grafica deve ormai esser considerata una vera e propria periferica che, collegata al computer, consente di disegnare in alta risoluzione e a colori.

In pratica la tavoletta grafica comunica al computer le coordinate della punta della penna che viene premuta sulla sua superficie che è grossomodo proporzionale a quella dello schermo TV.

Oltre alla tavoletta è ovviamente necessario il software per sfruttarla al massimo. In genere, insieme con la periferica, viene venduto il nastro (o il disco) contenente il programma da utilizzare. Alcuni software sono straordinariamente potenti perchè, oltre a consentire il tracciamento di semplici linee, permettono di realizzare con grande semplicità cerchi, ellissi, aree piene e vuote, duplicazioni di immagini, registrazioni su supporto magnetico, stampa su carta eccetera.

Per il C/64 sono commercializzate numerose tavolette, davvero semplici e divertenti da usare. Per il Plus/4 (ahinoi!) non c'è nulla in vendita e dubitiamo che qualche Ditta possa imbarcarsi in un'impresa del genere, commercialmente poco redditizia a causa del basso numero di utenti dello sfortunato computer.

Espansioni per C/16

☐ **Sul N.28 di C.C.C avete parlato di modifiche da apportare al C/16 per espanderlo. In che cosa consistono e come realizzarle?**
(Marco Garofalo, Modica)

• L'intervento non è semplice come credi dato che richiede una buona conoscenza di hardware, la necessità di procurarsi i circuiti integrati e di realizzare quelli stampati (tra l'altro, a doppia faccia).

Avevamo sentito parlare di alcune Ditte che erano in grado di apportare le modifiche o di procurare il necessario per farlo. In seguito, però non hanno insistito nel pubblicizzare le offerte. Hai provato, comunque, a contattare i nostri inserzionisti che vendono hardware per corrispondenza? Anche se nella loro pubblicità non compaiono questi apparecchietti, spesso sono in grado di accontentare le richieste più strane degli utenti Commodore!

Compatibilità di file

□ E' possibile usare gli archivi di "Superscript" (C/128) su "Easy Script" (C/64) e viceversa?
(Paolo Bonazzi, S.Lazzaro)

• Se un programma non prevede esplicitamente la compatibilità con altri programmi, è purtroppo necessario rinunciare all'impresa a meno che non si è bravi nel programmare e risalire, da soli, alla gestione dei singoli file.

Nel caso specifico è possibile (proprio perchè è previsto) leggere i file generati da Easy Script (E/S) se si lavora col programma Superscript (S/S). Il viceversa è anche possibile a patto che si rinunci a rendere operative alcune funzioni che, tipiche di Superscript, non sono gestibili con E/S. Non dimentichiamo, infatti, che quest'ultimo programma è nato quando del C/128 non si parlava nemmeno.

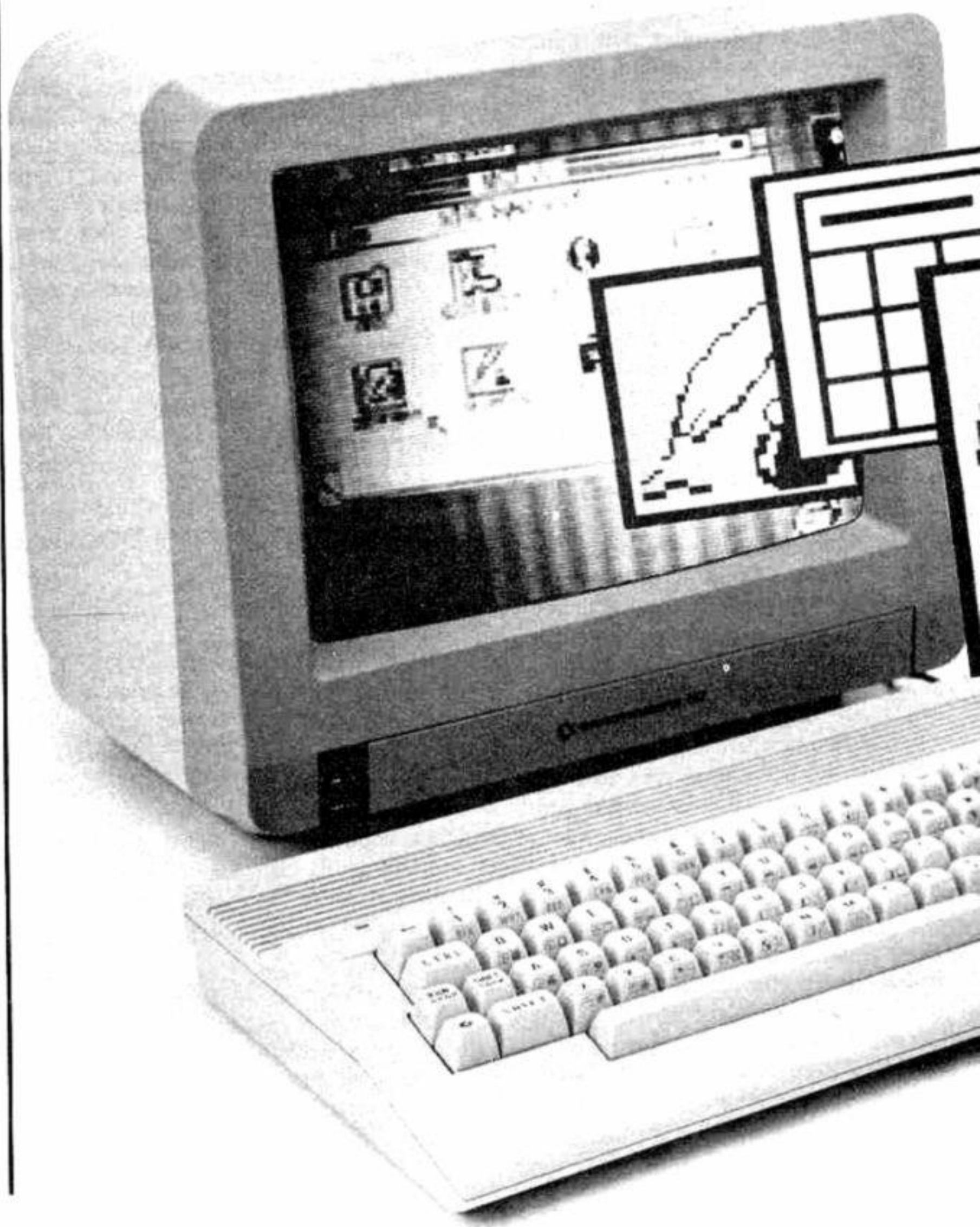
Dopo la diffusione del Commodore 128, la software house che produsse E/S pensò a realizzare S/S tenendo conto, saggiamente, della possibilità di leggere senza alcun problema file generati precedentemente con E/S.

Le Case di Software serie (quale è il caso della "Precison Software") pensano sempre ai propri affezionati clienti e, di norma, se creano un programma migliore (o, semplicemente, propongono una nuova versione), fanno sempre in modo che file generati col software precedente siano utilizzabili ancora. Il contrario, come è intuibile, non sempre è possibile realizzarlo proprio perchè, nella maggior parte dei casi, nelle versioni più recenti si introducono utility più potenti e, comunque, non presenti nelle precedenti "release".

Concludendo: prova a caricare, con E/S, file generati da S/S: se compaiono caratteri strani (specie se in Reverse) cancellali e sostituiscili con altri che conosci.

Nel caso di semplici documenti (lettere, appunti, racconti) non vi saranno grandi cambiamenti da apportare; nel caso di tabelle di grandi dimensioni (specie se formattate in modo molto particolare) sarà praticamente impossibile utilizzarle con Easy Script.

I grand

NUOVO 64, PIÙ

ommodore sono più grandi in tutto.

GRANDE NELL'ISTRUZIONE.

Poke per C/16 Plus/4

Numerosi lettori richiedono di indicare alcune Poke per la generazione di effetti "strani". Un nostro lettore, Paolo Lizio, ne propone parecchie e ne approfitta per ricordare brevemente il funzionamento dell'istruzione Poke.

Per chi non sa nulla

Esiste un comando per assegnare un determinato valore ad una locazione di memoria: POKE.

Grazie a questa istruzione è possibile ottenere diversi effetti: quelli che vi proponiamo "girano" sul Commodore 16 e sul suo fratello maggiore Commodore plus/4.

La corretta sintassi del comando è il seguente:

POKE(Locazione),(Valore)

Le locazioni di memoria sono numerate da 0 a 65535 (in esadecimale: da 0000 a FFFF), mentre il valore che si può assegnare ad esse deve essere compreso tra 0 e 255 (00, FF).

Con il comando PEEK, al contrario, si può invece sapere quale valore è assegnato ad una determinata locazione di memoria. Scrivendo PRINT PEEK (1344), seguito dalla pressione del tasto Return, il computer visualizzerà il valore assegnato alla locazione 1344.

In alcuni casi la modifica del contenuto di una locazione non sortisce alcun effetto particolare; in altri casi si ottiene un completo disastro. Nei casi... intermedi le conseguenze sono decisamente interessanti.

Ma bando alle ciance e passiamo alla descrizione di alcune situazioni interessanti:

POKE 240,1
Attende la pressione di un tasto; praticamente ha lo stesso effetto dell'istruzione GETKEY, con la differenza che può essere utilizzato anche al di fuori di un programma.

POKE 768,1
"Entra" nel programma Monitor (per lavorare in linguaggio macchina) e contemporaneamente disattiva il comando "X" per uscirne. Fino a quando non premerete il tasto Reset non potrete tornare in BASIC.

POKE 770,2
Niente paura, non si è rotto il computer! Per sbloccarlo è sufficiente premere il tasto di Reset.

POKE 772,22
Il computer risponde SYNTAX ERROR a tutti i comandi.

POKE 772,77
Identico al precedente, ma compare il messaggio OUT OF MEMORY ERROR.

POKE 772,44
Il computer si blocca appena cercherete di eseguire un comando. Premere Reset per lo sblocco.

POKE 774,255
Molto utile per proteggere il listato di un programma: il comando LIST non servirà molto...

POKE 774,110
Il listato è di nuovo a portata di mano.

POKE 786,68
Il tasto Run/Stop non funziona più!

POKE 786,66
Il tasto Run/Stop funziona nuovamente.

POKE 817,255
Sarà impossibile salvare su nastro o disco un programma: il comando SAVE non verrà più preso in considerazione.

POKE 817,241
Potete usare di nuovo il comando SAVE.

POKE 1344,1
Come saprete, il Commodore 16 è dotato di autorepeat: tenendo premuto un tasto il carattere corrispondente viene stampato di continuo fino al rilascio del tasto stesso. Con questa POKE verrà disabilitato l'autorepeat (ma continuerà a funzionare per lo spazio, i tasti cursore e Ins/del): continuando a tenere schiacciato un tasto, si otterrà sul video un solo carattere.

POKE 1344,128
Riabilita l'autorepeat.

POKE 2025,64
L'uso del tasto Insert è possibile fino a quando l'ultimo carattere della riga arriva alla fine della riga stessa.

POKE 2025,0
L'uso del tasto INSERT ritorna normale.

POKE 2026,255
Ha la stessa funzione di ESC+A: causa l'inserimento automatico di caratteri all'interno di una riga.

POKE 2026,0
Disabilita l'inserimento automatico.

POKE 2039,1
Rende impossibile il blocco dello scrolling dello schermo effettuato con i tasti CONTROL e S.

POKE 2039,0
Riabilita CONTROL+S.

POKE 65286,64
Blank dello schermo: questo viene nascosto e compare dello stesso colore del bordo, potete scrivere con la tastiera, ma non vedrete niente sul video.

POKE 65286,27
Per potere rivedere lo schermo dovete scrivere "alla cieca" questa istruzione.

POKE 65287,32
Il computer si... spegne e non c'è verso di riaccenderlo!

POKE 65287,136

Questa è forse l'istruzione più utile di questo elenco: come certamente saprete, il Commodore 16 può scrivere in due diversi modi, selezionabili premendo contemporaneamente i tasti Shift e Commodore: maiuscolo/grafico e minuscolo/maiuscolo. Grazie a questa POKE i due modi si possono "fondere": si scrive, normalmente, in modo maiuscolo/grafico ma, attivando il REVERSE (tasti CTRL + 9), invece delle lettere in negativo, compariranno le lettere minuscole! Provare per credere...

POKE 65287,8

Disabilita la POKE precedente.

POKE 65290,0

Ottima per fare uno scherzo ad un amico: lo schermo appare normale, con il cursore lampeggiante, ma nessun tasto funziona!

E per finire, quattro routine del linguaggio macchina richiamabili con il comando SYS:

SYS 32768

Reset parziale del computer, ovvero senza cambiare i colori del bordo e dello schermo.

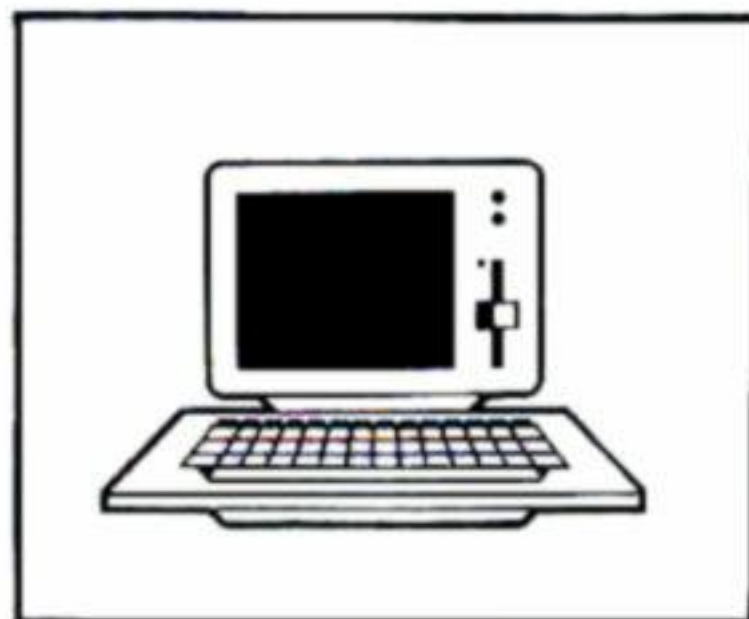
SYS 65529

Reset totale del computer: lo schermo appare come all'accensione.

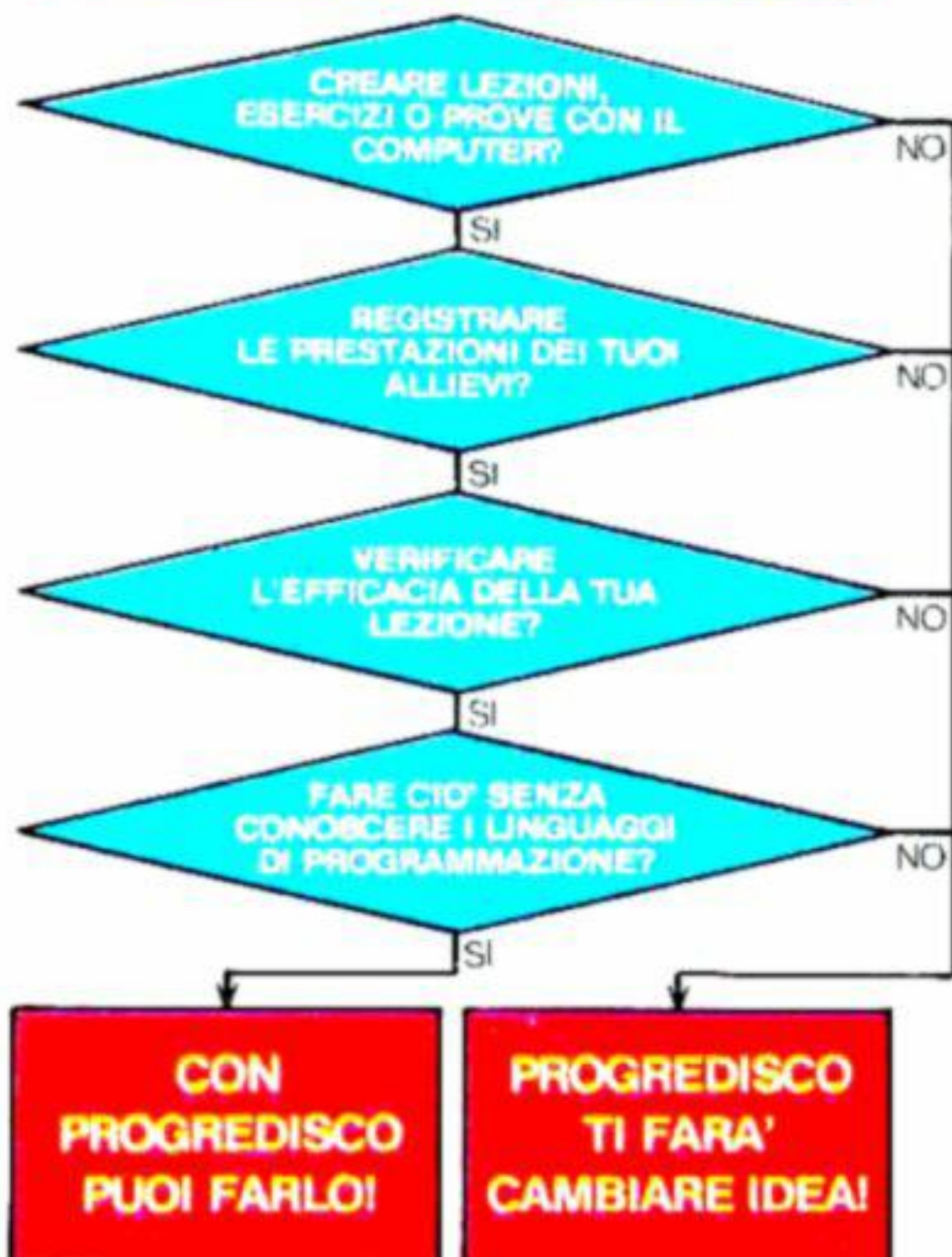
SYS 65499

Azzera l'orologio del computer (variabili TI e TI$).

SYS 65511

Chiude tutti i files precedentemente aperti.

Per i più... informati

Oltre, naturalmente, a quelli pubblicati finora su C.C.C., avete scoperto qualche Poke o qualche Sys dagli effetti interessanti?

Inviatene (almeno cinque per volta) e, se meritevoli, pubblicheremo le vostre scoperte.

C 64 - C 128

Rasputin

Questo gioco Mastertronic, che si carica in 128 giri di nastro, appartiene alla nuova serie di videogame "misti" in cui, cioè, risulta necessaria non solo abilità nel manovrare il joystick, ma anche intelligenza nell'effettuare le scelte opportune.

Con un sottofondo musicale di motivi russi (nella schermata c'è addirittura il titolo in caratteri cirillici), dovete dirigere il valoroso guerriero alla ricerca (e distruzione) del Gioello dei Sette Pianeti.

Si hanno a disposizione poteri magici

(da usare con saggezza per non sprecarli) che aumentano se rintracciate le pietre col marchio di Rasputin. Il gioco prosegue, di schermata in schermata, a patto che riusciate a raggiungere gli otto simboli che compongono il nome "Rasputin".

In basso, sul video, compaiono costantemente alcune icone (immagini) che hanno il compito di tenervi aggiornati sulla situazione del gioco.

Simpatica la musica (escludibile); ben fatta la grafica che riproduce, in tre dimensioni, il labirinto da percorrere, costituito da file di mattoni sovrapposte.

Countdown to Meltdown

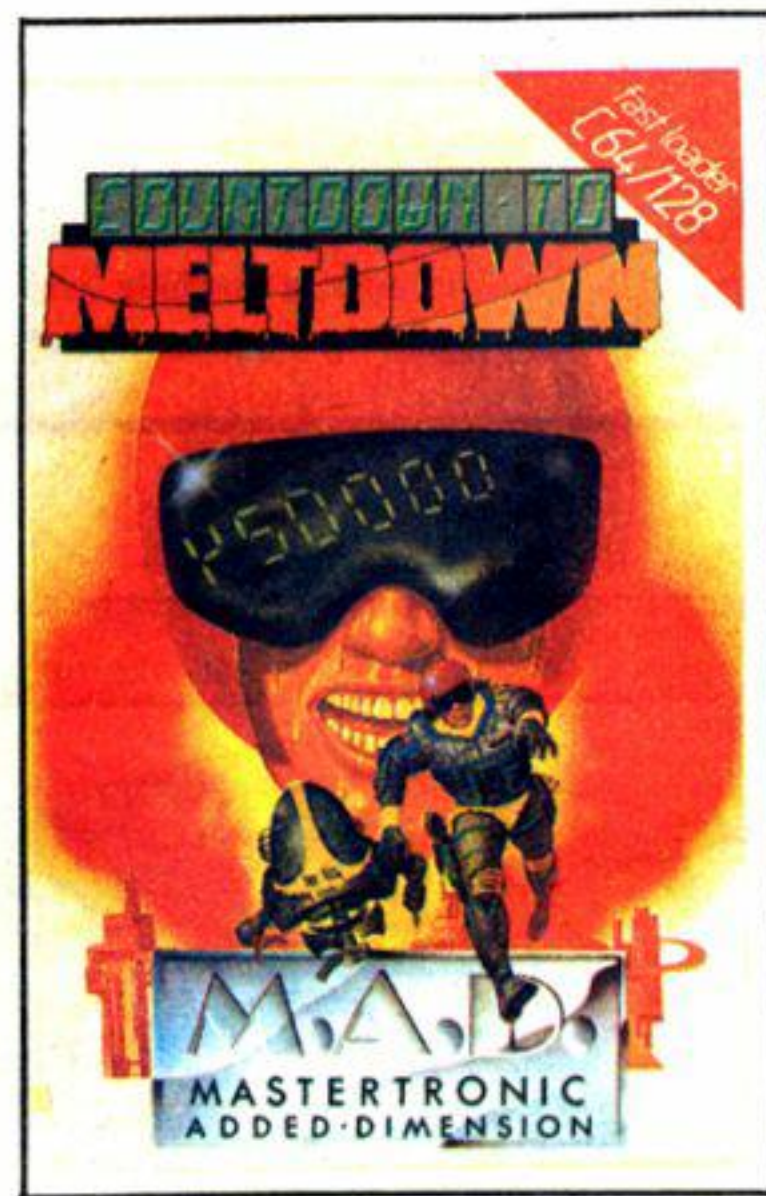
80 giri di nastro richiede questo videogame che, come tanti altri della Mastertronic, segue il filone dell'Adventure "animato". Questo, infatti, consiste nell'attraversare numerose "stanze" che, almeno stando al foglio allegato, sono più di 2000. Naturalmente non le abbiamo visitate tutte e ci limitiamo a dire che sono graficamente ben fatte.

Le schermate simulano la visione che restituirebbero le telecamere installate in alto su ciascuna stanza. Il vostro compito, infatti, consiste, utilizzando la rete di telecamere, nel guidare alcuni androidi alla ricerca di un guasto occorso in una centrale nucleare che sta addirittura per esplodere, mentre una miriade di informazioni sono costantemente presenti nelle ultime tre righe dello schermo.

Vi sono porte da aprire, passaggi da attraversare, armadi da esaminare; il tutto mentre siete inseguiti da strane forme viventi.

Una certa emozione ha suscitato l'esame del gioco che, per puro caso, è stato provato proprio nel periodo del disastro nucleare di Chernobil in Unione Sovietica...

Come ogni adventure che si rispetti, è possibile registrare su nastro una "situazione" per poterla riprendere successivamente.

Consigliato agli appassionati di Adventure dovrebbe esser esaminato anche dai patiti dei giochi in cui il joystick fa da padrone. Se non altro per verificare che, oltre ad esser lesti di mano, si è ancora lesti col cervello...

Wing Commander

Molto bello questo gioco/simulazione della Mastertronic nonostante siano sufficienti solo 57 giri di nastro per il suo caricamento.

La "trama" è piuttosto banale, ma non per questo il gioco è privo di interesse.

Siete pilota di un caccia a reazione e vi è stato affidato il compito di difendere l'isola, su cui vi trovate, da un attacco nemico. Dovete alzarvi in volo ed esplorare il cielo alla ricerca dell'avversario per abbatterlo con missili o colpi di mitragliatrice.

A differenza dei soliti giochi, in questo caso tutte le azioni sono aderenti alla realtà. I missili sono numerati e così pure i colpi del mitra e non è possibile sparare all'infinito; se la velocità è troppo bassa e tentate di puntare il muso verso l'alto, l'aereo cabra attivando un segnalatore acustico di pericolo; se la velocità è eccessiva la carlinga inizia a vibrare (riportando tale effetto sul video) e, se non siete lesti nel decelerare, potete riporta-

re danni gravissimi. Anche nella partenza e nell'atterraggio è necessaria una certa cura (è presente perfino il comando per far rientrare il carrello!) per evitare decolli e atterraggi disastrosi.

Durante il volo lo schermo è diviso in due parti di cui quella superiore simula la vista attraverso la calotta.

Nella parte inferiore, ampia metà schermo, sono presenti i segnalatori di benzina, potenza, posizione del carrello, entità dei danni riporatati, altitudine, velocità e tutte le indicazioni, insomma, utili per la corretta gestione delle manovre.

Un discorso a parte merita il doppio Radar. Il primo rappresenta la porzione di territorio sorvolato, mentre il secondo indica la posizione dell'eventuale aereo nemico.

L'intera isola, i suoi insediamenti, la nostra posizione, quella del nemico ed il suo obiettivo, possono esser esaminati con la pressione del tasto P. Oltre al joystick, infatti, è necessario ricorrere alla pressione di numerosi tasti per una completa manovrabilità del gioco.

Wing Commander, in conclusione, è una simulazione a tre livelli ben fatta, emozionante e, soprattutto, piuttosto aderente alla realtà (anche se tale affermazione è fatta da chi non ha mai pilotato un aereo...)

The Ultimate

Sul numero scorso ci siamo già occupati della Ditta Niwa di Sesto San Giovanni (Tel. 02/2440776) presentando lo Speed Dos.

Stavolta parliamo di una cartuccia, da inserire sul retro del C/64 oppure C/128, dalle caratteristiche davvero interessanti. Ci limiteremo a descrivere la maggior parte dei comandi che, ne siamo certi, saranno apprezzati da chi desidera "spremere" il proprio computer senza, però, manometterlo in alcun modo.

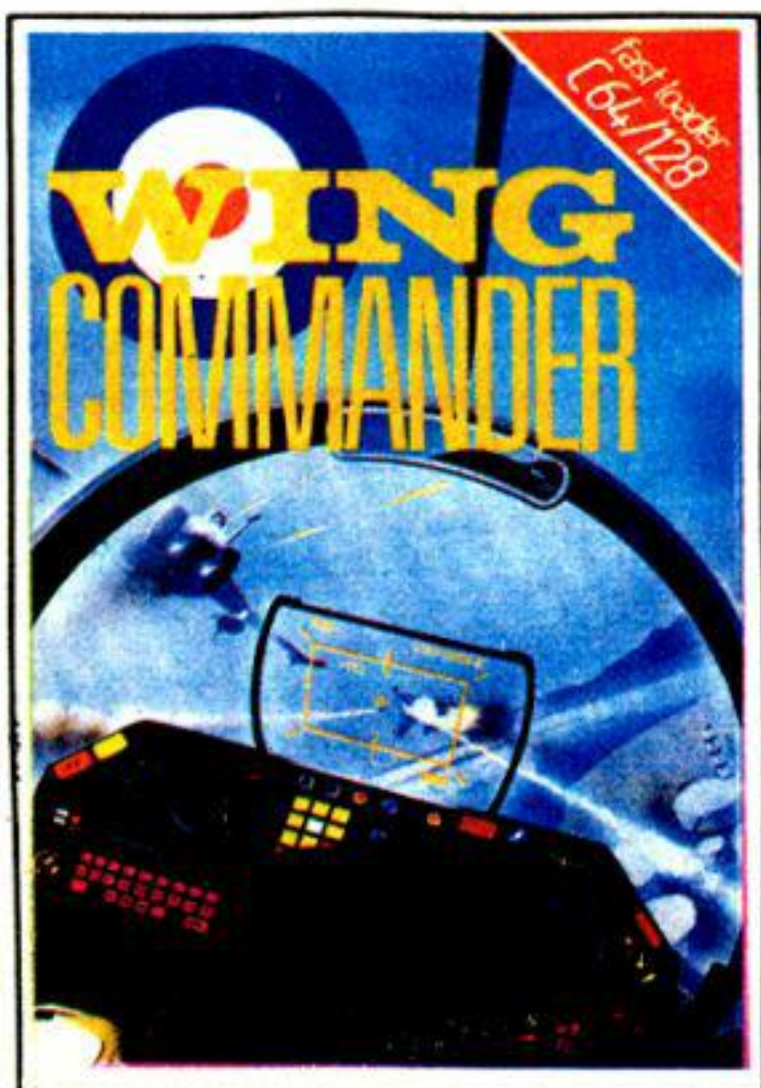
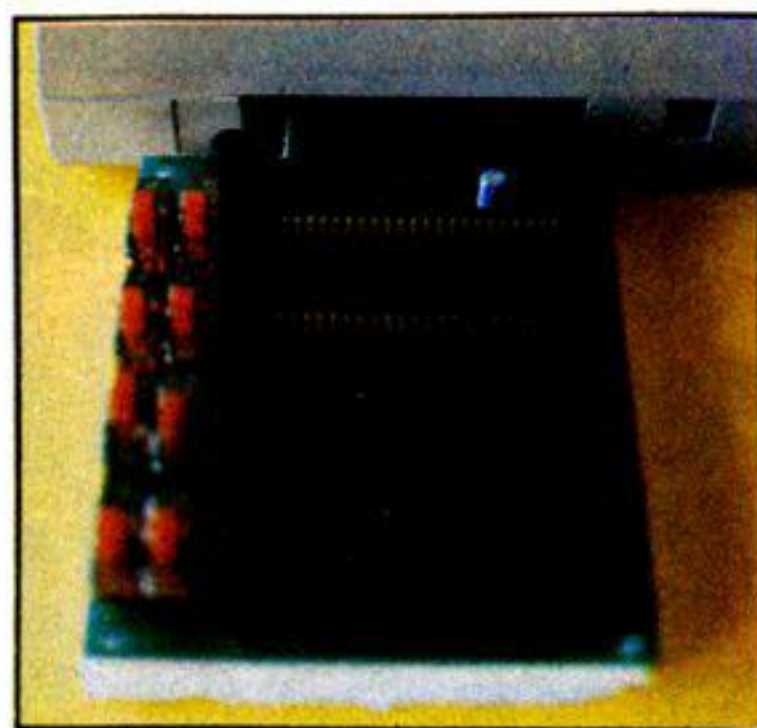
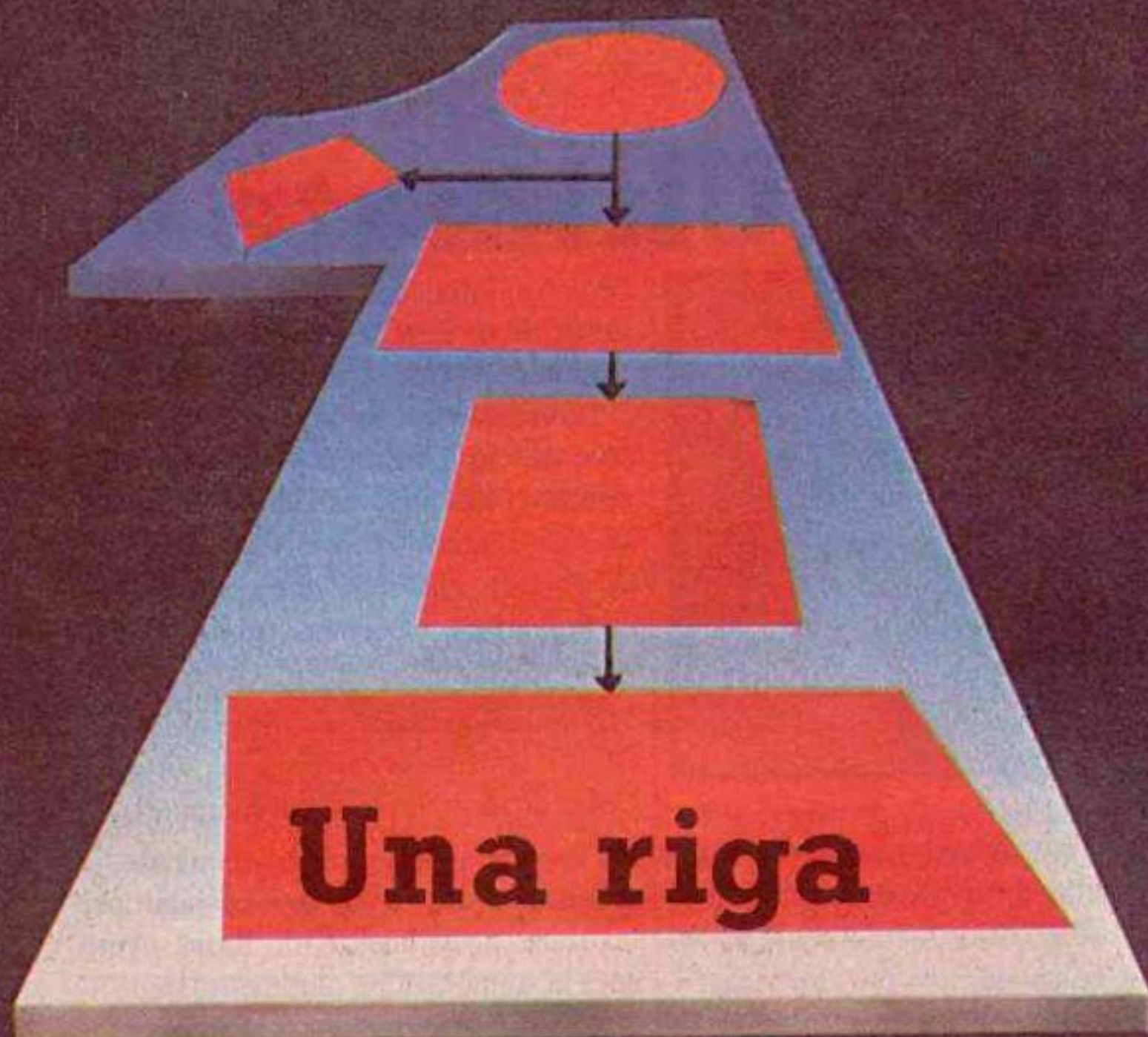
Non appena si accende il calcolatore, una schermata consente di scegliere tra ben nove opzioni. Alcune permettono operazioni sui file (copia da/su disco/nastro, turbo tape, turbo disk); non manca la visualizzazione della directory nè la formattazione veloce oltre alla facilitazione di tutte le operazioni su disco (rename, scratch, initialize, eccetera).

Interessante il sofisticato monitor per lavorare in Linguaggio Macchina (Supermon) rilocabile dall'utente. I comandi di Supermon sono tutti quelli necessari per lavorare senza problemi sia in linguaggio macchina "puro" che in Assembler.

Il comando Quit consente di escludere completamente la cartuccia via software evitando di rimuoverla dal computer.

Oltre alle opportunità esaminate, è possibile disporre di nuovi comandi per lavorare agevolmente in Basic. Auto (per la numerazione automatica delle linee Basic); Renum (per la renumerazione, compresi i Goto e Gosub); Del (cancellazione linee); Find (per rintracciare linee contenenti specifiche istruzioni da indicare nel comando); Merge (per "fondere" due programmi); Help (per individuare l'errore occorso); Dump (per esaminare le variabili adoperate); Trace (per esaminare lo svolgimento del programma, istruzione dopo istruzione); Hex, Dec (per le conversioni decimali esadecimali); Old (per recuperare un programma Basic accidentalmente cancellato con New).

Portacartucce

Un accessorio indispensabile (anche se un po' caro...) proposto dalla Ditta Niwa di Sesto San Giovanni. Quattro slot per altrettante cartucce da mantenere in permanenza sul retro del computer. L'alimentazione elettrica lascia "accese" tutte le cartucce inserite che possono esser selezionate, una alla volta, da due interruttori che agiscono, a seconda dei casi, sul pin che le rende "visibili" al calcolatore. In caso di pasticci niente paura: un pulsante di Reset incorporato ripristina il normale funzionamento

Su ogni numero di Commodore Computer Club compaiono un paio di pagine dedicate a una dozzina di micro programmi lunghi una sola riga.

E' ovvio che non è possibile pretendere effetti sorprendenti in listati così brevi, tuttavia, sviluppando l'idea su cui gli stessi programmi son basati è possibile pervenire a realizzazioni di tutto rispetto ricorrendo, è inutile dirlo, a qualche riga in più.

Particolarmente interessanti per i principianti, spesso utili anche per gli esperti, i micro programmi di una sola riga rappresentano una valida "palestra" per abituarsi a concentrare in poche, essenziali istruzioni la soluzione di problemi complessi solo in apparenza.

Per Commodore 128

Ecco una decina di righe specifiche per C'128.

Appaiono notevolmente più lunghe delle righe per C/64, e ciò è dovuto al fatto che il C/128 accetta ben 160 caratteri per linea "logica" contro gli 80 del C'64 e del C'16.

I minilistati per C'128 sono opera di Dario Villone, un nostro lettore di Torino.

1

Contasecondi. Il computer chiede quanti secondi deve contare. Dopo l'inserimento si udrà un segnale sonoro.

Alla fine del conteggio se ne udrà un altro ma di tono più alto. Per uscire dal programma è sufficiente premere il tasto Return a vuoto.

```
1 INPUT"SECONDI";N:
  IFNTHENSOUND1,4000,20:
  SLEEPN:SOUND2,8000,20:
  N=0:GOTO1
```

2

Somma e media. Il computer è pronto per ricevere una serie di numeri che sommerà diligentemente e dei quali calcolerà la media non appena verrà premuto un Return a vuoto.

```
1 PRINT"SOMMA (0=FINE)"
  :DO:A=0:INPUTA:T=T+A:
  N=N+1:LOOPUNTILA=0:
  PRINT"TOTALE =";T:
  M=N-1:PRINT"ADDENDI="
  ;M:IFMTHENPRINT"MEDIA
  ARITMETICA =";T/M
```

3

Sconto. Non fatevi attrarre da tutte le offerte di sconti e di saldi di fine stagione!!

Ma se proprio non ne potete fare a meno questa riga vi aiuterà a determinare il reale sconto applicato conoscendo il prezzo di listino e la cifra pagata.

```
1 PRINT"SCONTO":INPUT"
  QUAL'ERA IL PREZZO";P:
  INPUT"QUANTO HAI
  PAGATO";S:PRINT
  "SCONTO APPLICATO:";:
  PUDEF" .,L":
  PRINTUSING"##.##";
  (P-S)*100/P;:PRINT" %"
```

4

Viaggio. Questo miniprogramma si rivela utile per calcolare in anticipo la durata di un viaggio conoscendo la velocità media e la lunghezza del percorso. Per terminare battere il consueto Return a vuoto.

```
1 PRINT"VIAGGIO":V=0:
  INPUT"VELOCITA' MEDIA
  IN KM/H";V:IFVTHEN
  INPUT"DISTANZA IN KM";D:
  PRINT"TEMPO: ";:
  PRINTUSING"###";(D/V)*60;:
  PRINT" MINUTI":GOTO1
```

5

Contaminuti. Un pratico e preciso contaminuti basato sull'uso di "Sleep", una potente istruzione del Basic 7.0 che genera una pausa in secondi in funzione del numero che la segue. (Es. Sleep 10 genererà una pausa di 10 secondi).

```
1 INPUT"QUANTI MINUTI";N:
  IFNTHENSOUND1,4000,20:
  SLEEPN*60:FORI=1TO20:
  SOUND2,1,5:
  SOUND2,12000,5:
  NEXT:N=0:GOTO1
```

6

Stampa listato e directory. Una sola riga per automatizzare la procedura di stampa su carta dei listati e delle directory.

Dopo il Run, per stampare il programma in memoria sarà sufficiente premere il tasto F8. Per caricare e stampare una directory bisognerà premere in successione i tasti F4 e F8. (Attenzione: così facendo si perde il programma in memoria).

```
1 KEY8,"OPEN4,4:CMD4:
  LIST:PRINT#4:CLOSE4"
  +CHR$(13):KEY4,"LOAD"
  +CHR$(34)+"$"+CHR$(34)
  +",8"+CHR$(13):
  PRINT"STAMPA LIST = F8"
  :PRINT"STAMPA DIRECTORY
  = F4 E POI F8"
```

7

Cronometro. Anche se non possedete un mega-orologio con 99 e più funzioni (ivi compreso un cronometro) non disperate: questa riga, dopo il Run, visualizzerà un preciso cronometro utile in svariate occasioni.

```
1 TI$="000000":DO:
  H$=LEFT$(TI$,2):
  M$=MID$(TI$,3,2):
  S$=RIGHT$(TI$,2):SCNCLR:
  PRINT"ORE: ";H$:
  PRINT"MINUTI: ";M$:
  PRINT"SECONDI: ";S$:
  SLEEP1:GETA$:
  LOOPUNTILA$<>""
```

8

Segnale acustico. Una riga che, oltre a dare un saggio delle possibilità sonore del C/128, si rivela utile nel caso doveste inserire un segnale acustico all'interno dei vostri programmi.

Il segnale rimarrà memorizzato sul tasto funzione F1.

L'istruzione "Sound1,1,1" serve ad eliminare il rumore di fondo senza ridurre il volume e senza il fastidioso "Click".

```
1 PLAY"V1O5T7U9X0CEGD#
  FACEGC":SOUND1,1,1:
  KEY1,"PLAY"+CHR$(34)+
  "V1O5T7U9X0CEGD#FACEGC"
  +CHR$(34)+":
  SOUND1,1,1":PRINT"SEGNA
  LE ACUSTICO ATTIVATO SU F1"
```

9

Paga Colf. Se la nostra Colf (ovvero Collaboratrice familiare) percepisce ad esempio 6000 lire all'ora e oggi ha lavorato dalle nove e quindici a mezzogiorno meno cinque, quanto le dovremo dare ?

Il programma lo calcola per voi.

```
1 INPUT"INGRESSO: ORE,
  MINUTI";O,M:
  INPUT"USCITA: ORE,
  MINUTI";P,Q:
  INPUT"PAGA ORARIA";S:
  PRINT"PAGA: ";:
  PUDEF" .,L":
  PRINTUSING"##,###";
  ((P-O)*60+Q-M)*S/60
```

Per Commodore 64

Ecco ora una manciata di righe dedicate ai "sessantaquattristi".

I cinque programmini sono opera di Paolo Valvo di Pachino (SR).

10

Bordo bicolore. Questo minilistato crea un interessante effetto cromatico sulla cornice del vostro schermo: avrete infatti due strisce colorate in movimento.

```
1 FORJ=0TO1:
  POKE53280,J:
  NEXT:GOTO1
```

11

Campana. Sfruttando il SID ecco un interessante effetto sonoro che riproduce il suono di una campana.

```
1 S=54273:POKES+23,15:
  POKES,158:POKES+4,9:
  POKES+14,30:POKES+3,21:
  FORJ=0TO400:NEXT:
  POKES+3,20:RUN
```

12

Quadrati colorati. Lo schermo del vostro 64 verrà cancellato da una cascata di quadratini colorati.

```
1 PRINTCHR$(147):FORJ=0TO999:
  POKE1024+J,160:
  POKE55296+J,RND(1)*255:NEXT:
  RUN
```

13

Arcobaleno. Il ricorso al Linguaggio Macchina apre possibilità notevoli soprattutto nel campo degli effetti speciali.

Questa riga (appunto in L.M.) crea un effetto assolutamente impensabile con il Basic.

Nota Bene

Alcune righe tra quelle pubblicate sembrano possedere più di 80 caratteri e, come tali, inaccettabili dal computer.

Nei casi in cui ci si accorga che la riga è troppo lunga, è necessario ricorrere alle abbreviazioni dei comandi così come indicato nell'appendice specifica riportata nel manuale del computer in vostro possesso.

Ad esempio invece di scrivere PRINT è possibile abbreviare col punto interrogativo (?). Invece di POKE potete scrivere il carattere "P" seguito dal carattere che viene visualizzato premendo contemporaneamente il tasto shift insieme con "O". Tutte le abbreviazioni possibili, lo ripetiamo, sono riportate in una delle appendici di qualsiasi manuale Commodore.

Nel caso sbagliate a digitare i microlistati che superano, in lunghezza, gli ottanta caratteri (SYNTAX ERROR o altri tipi di errore), è necessario, per sicurezza, ribatterli per intero e non apportare modifiche alla riga visualizzata con l'istruzione LIST

La partecipazione dei lettori è gradita e compensata, in caso di pubblicazione, con materiale della Systems editoriale (libri, fascicoli arretrati, abbonamenti, eccetera).

Inviate le vostre 1 RIGA, purchè nella misura di almeno dieci per volta. Non ci è infatti possibile raggruppare i (moltissimi) micro-programmi che pervengono singolarmente in Redazione.

Inviate i vostri lavori su carta, (meglio se su nastro) corredati, ciascuno, di una breve spiegazione sulla funzione che compiono, proprio come li vedete pubblicati in queste pagine.

Ricordate di indicare chiaramente nome, cognome, via e città, e indirizzate a:

Commodore Computer Club
SYSTEMS EDITORIALE
Rubrica "1 Riga"
Viale Famagosta, 75
20142 MILANO

```
1 FORJ=0TO9:READA:
  POKE 828+J,A:NEXT:
  SYS828
  DATA 120,206,33,208
  ,206,32,208,76,61,3
```

14

Parabola. La scritta "una riga" verrà stampata secondo una parabola. Variando la stringa sarà possibile ottenere una titolatrice per i vostri programmi.

```
1 PRINTCHR$(147):A=7/30
  FORJ=1TO5STEPA:
  POKE 214,25-J*J:
  POKE211,J/A:SYS58732:
  PRINT"UNA RIGA";:NEXT
```

Per Commodore 16 e 128

Ecco alcune righe per C'16 (valide anche per il C'128) ideate da Franco Rossi di Udine.

15

Quadro d'autore. Con il comando Circle si ottengono effetti grafici degni, appunto, di un quadro d'autore.

```
1 COLOR 0,1:
  COLOR1,16:
  GRAPHIC1,1:
  FORA=1TO800STEP2:
  CIRCLE1,150,100,A,A-A,,,A,255:
  NEXT
```

16

Effetto Circle. Un'altra riga che genera un fantastico effetto grafico.

```
1 GRAPHIC1,1:
  FORA=1TO160 STEP2:
  CIRCLE 1,150,100,A,A,A,,,A:
  NEXT:GETKEYA$:
  GRAPHIC0
```

17

Scritta scorrevole. Con un intelligente utilizzo del comando CHAR si ottengono effetti interessanti come questo.

```
1 PRINTCHR$(147):
  FORI=26TO0STEP-1:
  FORA=1TO80: NEXTA:
  CHAR1,I,8,"SCORRE ":
  NEXTI:GOTO1
```

18

Spirale Box. Un altro effetto grafico di notevole interesse ottenuto mediante il comando Box.

```
1 GRAPHIC1,1:
  FORA=1 TO300STEP2:
  BOX1,300,250,A,35,A:
  NEXTA:
  GETKEYA$:GRAPHIC0
```

A tutti i Commodore Computer Club

Molti circoli si sono aperti e molti sono usciti allo "scoperto" dopo il nostro invito ad aprire un Computer Club, apparso sul N. 21 di C.C.C.

Allo scopo di rendere un servizio migliore ai nostri lettori che intendano contattare uno di questi simpatici circoli culturali, i segretari dei circoli stessi sono pregati di compilare il seguente tagliando, o sua fotocopia, e di inviarlo in busta chiusa (affrancata secondo le vigenti tariffe postali) a:

Systems Editoriale
Servizio Notizie Computer Club
Viale Famagosta, 75
Milano
La completa compilazione dell'intera scheda, è INDISPENSABILE per la pubblicazione gratuita sulla nostra rivista.

Nome del club: .... Sede del club: .... Via ....
C.A.P. .... Città .... Prov.: .... Tel. Prefisso: ...... N. ....
Presidente: .... Segretario: .... N. soci fondatori: ....
N. di soci finora iscritti: .... Data di fondazione: .... Giorni di apertura della sede: ....
Orario di apertura: .... Computers disponibili (specificare): ....
Periferiche disponibili (specificare): .... Programmi disponibili (N. approssimativo): ....
Videogiochi N.: .... Professionali N.: .... Biblioteca tecnica N. volumi: ....
N. abbonamenti a riviste italiane: .... N. abbonamenti a riviste straniere: .... Quota di iscrizione L. ....
(Specificare se annuale, mensile ecc.) ....
Attività previste: .... Bollettino periodico emesso: .... Attività già svolte: ....
Eventuali sponsor: .... Disponibilità alla sponsorizzazione (sì/no): ....

Il sottoscritto ...., presidente del Computer Club .... autorizzo la Systems Editoriale a diffondere notizie riguardanti le attività del circolo culturale citato anche se pervenute in redazione in via non ufficiale.
Dichiaro inoltre che le informazioni comunicate corrispondono al vero e che, in caso di difformità accertata da parte di incaricati della Systems stessa, Commodore Computer Club, allo scopo di tutelare la buona fede degli utenti della rivista, si riserva il diritto - dovere di avvertire i propri lettori nel modo e nella forma che riterrà più opportuni.

In fede

La gestione dei quattro banchi di memoria

Dove (e come) conviene posizionare la pagina grafica? E, i caratteri ridefiniti, che fine fa il programma Basic? E altre cose ancora...

di Pasquale D'Andreti

In questo terzo articolo dedicato alla descrizione dei possibili modi grafici di cui il C/64 dispone, tratteremo approfonditamente dei problemi che sorgono allorquando si seleziona un banco di memoria diverso da quello in uso normalmente.

Discuteremo anche sulle precauzioni da osservare affinchè non si creino conflitti tra il programma Basic ed eventuali pagine grafiche.

Infine, come necessario compendio di questa serie di articoli, provvederemo ad esporre una dettagliata tabella riassuntiva, resa sotto forma di un breve programma in Basic, in cui sono contenute le informazioni indispensabili per raggiungere tutti i modi e tutte le configurazioni grafiche possibili.

Prima di proseguire, facciamo presente che il testo che segue è stato redatto in modo da non richiedere particolari conoscenze di base; cionondimeno, si consiglia la (ri)lettura della prima (CCC N.28 "Grafica in alta risoluzione") e della seconda parte della serie (CCC N.30 "Ridefinizione dei caratteri"), quel tanto che basta per entrare nello spirito della trattazione.

La selezione dei banchi di memoria video

Come affermato negli articoli accennati, il Vic II (nome dell'integrato 6567 che gestisce la costruzione "fisica" dell'immagine video) può "vedere" soltanto 16K byte di memoria per volta: ciò vuol dire che qualsiasi tipo di pagina grafica intendiamo utilizzare, sia essa in alta risoluzione che in modo testo, la dovremo posizionare entro un'area di memoria la cui ampiezza si estende per 16384 byte.

Questa non è affatto una limitazione ma, anzi, costituisce un certo vantaggio in quanto i registri da manipolare per accedere ad una certa funzione sono più "corti" in numero di bit. Inoltre in 16K entrano due pagine grafiche in alta risoluzione oppure 16 pagine in modo testo.

Come tutti sapranno, il C/64 dispone di 65536 byte RAM (numerati da 0 a 65535) da cui si deduce che i banchi da 16K disponibili sono quattro.

Volendo modificare l'allocazione del banco usato, bisogna manipolare opportunamente i bit di posizione 0 e 1 del byte locato in 56576. La tabella N.1 indica i valori ed il loro preciso significato nonchè gli argomenti che l'istruzione POKE deve avere per realizzare l'effetto desiderato.

Nella suddetta tabella il valore da assegnare alla variabile B (che compare nel comando Poke) si deve prelevare dalla colonna a sinistra dove è indicato il numero progressivo del banco.

Come si può notare dalla tabella 2, il computer provvede a riservare il Kbyte utilizzato dalla memoria di schermo nelle locazioni che vanno dalla 1024 alla 2047.

Modificando, come descritto nella prima parte, la zona di memoria dedicata allo schermo in modo testo, si va incontro a conflitti di memoria tra la pagina grafica ed il sistema operativo che molto spesso terminano in un tragico ed irreversibile blocco del computer che costringe ad operare un reset oppure a spegnere e riaccendere il computer.

Questi conflitti possono avvenire quando la memoria video si sposta dalla sua allocazione originale all'area riservata al sistema operativo (0-1023) o alla zona dedicata all'immagazzinamento dei programmi Basic.

Nel primo caso, l'unica cosa da fare è tornare sui propri passi in quanto non è possibile escludere o spostare le funzioni del sistema operativo; nel secondo, invece, lo spostamento della pagina grafica in modo testo è possibile, a patto che si impedisca al Basic l'accesso a quella zona di memoria da noi prescelta.

Tabella N.1

Banco	Bit 0 e 1 di 56576	Zona RAM in uso
0	00	49152-65535
1	01	32768-49151
2	10	16384-32767
3	11	0-16383

```
POKE56576,(PEEK(56576)AND252)+B
```

A questo punto sorgono i problemi a cui cercheremo di dare una soluzione specifica: infatti ogni banco presenta difficoltà che sono assenti negli altri e viceversa.

Proprio per affrontare meglio questo discorso, suddivideremo la trattazione in quattro paragrafi nei quali mostreremo le mappe di memoria relative ai singoli banchi, indispensabili per la comprensione immediata del testo.

Il banco N.3

La tabella N.2 rappresenta, schematicamente, la mappa di memoria delle locazioni facenti parte del banco N.3, ossia quello utilizzato normalmente dal Vic II non appena si accende il computer.

Prima di continuare nel discorso è utile fare una breve sortita nel campo dell'immagazzinamento dei programmi e delle variabili Basic.

Appena il computer viene acceso, l'interprete Basic riserva le locazioni che vanno da 2048 a 40959 per i programmi e per le variabili; sottraendo la prima locazione dalla seconda (40959-2048) si ottiene la quantità di memoria disponibile, che è di 38911 byte, quantità che infatti

Tabella N.2

Area riservata al KERNAL al Basic e al 6510	0 1023
Memoria di schermo in condizioni normali	1024 2047
Area RAM riservata ai programmi ed alle variabili Basic	2048 16383

compare nel messaggio di "apertura" al momento dell'accensione.

Questa zona è controllata, come affermato in precedenza, dall'interprete Basic che, ad ogni operazione compiuta su di un programma (cancellazione o aggiunta di linee, esecuzione, lettura da disco o cassetta, modificazione o creazione di variabili) cambia il contenuto di alcune locazioni nell'ambito di questa memoria. Tali cambiamenti possono disturbare un'eventuale pagina grafica sita anch'essa in questo "range".

Ma non è questo il problema più grave: infatti se noi, disegnando in alta risoluzione o scrivendo qualcosa in modo testo, modifichiamo il contenuto della memoria, rischiamo, come minimo, di alterare nome e valore delle variabili, fino ad arrivare ad un "crash" del sistema.

Fortunatamente ciò si può evitare in due modi diversi la cui scelta dipende solo dal tipo e dalla lunghezza del programma.

In pagina zero vi sono due puntatori, ed esattamente la coppia locata in 43-44 e quella in 55-56, che indirizzano, rispettivamente, l'inizio e la fine della zona riservata al Basic.

Il primo si utilizza quando il programma che gestisce la pagina grafica è molto lungo oppure si ritiene che possa esser suscettibile di ulteriori ampliamenti. Per utilizzare questo sistema si "sposta" il Basic facendo puntare, subito dopo l'accensione del computer, le locazioni 43-44 appena dopo la fine della pagina grafica (−1 byte) e ponendo a zero la locazione immediatamente precedente.

C'è da osservare che questo primo metodo è alquanto fastidioso da usare in quanto bisogna digitare le istruzioni relative al posizionamento dei puntatori "a mano" prima di leggere (o digitare) il programma Basic ma, utilizzato quando si sceglie il terzo blocco, lascia, nel peggiore dei casi, 24575 byte liberi, che non sono affatto pochi.

Facciamo un esempio pratico: poniamo di voler utilizzare una pagina in alta risoluzione (per ulteriori dettagli si riveda il paragrafo dedicato nel primo articolo, CCC N.28) situata nel banco N.3 a partire dalla locazione 8192 che, richiedendo 8000 byte (numerati da 0 a 7999), "terminerà" nella locazione 16191 (8192 +7999). L'inizio del Basic dovrà essere spostato, da 2048, all'indirizzo 16193 (16192 +1) e la locazione 16192 dovrà essere posta a zero.

Dall'indirizzo 16193 si dovranno ricavare due valori da porre nelle locazioni 43 e 44; questa semplice (!) linea Basic compie il lavoro suddetto:

```
POKE44,16193/256:POKE43,16193-
(PEEK(44)*256):POKE16192,0:NEW
```

La linea deve essere digitata PRIMA di digitare o leggere il programma da disco o da nastro. Omettendola, i risulati saranno molto poco apprezzabili.

L'unico metodo per evitare ciò consiste nell'aggiunta, all'inizio del programma, di linee sul tono di queste:

```
0 IF PEEK(44)=INT(16193/256)
THEN GOTO 10
1 PRINT"HAI DIMENTICATO
DI DIGITARE .......": END
10 REM qui inizia il programma
.....(eccetera)...............
```

Al posto dei puntini, nella PRINT della riga 1, bisogna porre la linea citata prima.

Ricordiamo che quelli appena descritti sono esempi e che al posto delle costanti 16193 e 16192 bisogna porre i valori consecutivi desiderati, sapendo che il primo rappresenta l'inizio vero e proprio del programma e il secondo il byte precedente.

Se si intende utilizzare un programma breve, si può modificare il solo puntatore 55-56.

Questo deve essere cambiato in modo da indirizzare all'inizio della pagina grafica (meno un byte).

A differenza del primo caso, si può modificare questo puntatore anche da programma, a patto che ciò sia fatto nella sua prima linea in quanto l'abbassamento del limite superiore del Basic provoca la perdita di variabili alfanumeriche eventualmente elaborate.

Considerando l'ultimo esempio, e nel caso che il programma "post-run" (ossia con tutte le variabili utilizzate) non richieda più di 6K, posizioneremo il vettore 55-56 in modo da farlo puntare alla locazione 8191 (inizio della pagina in alta risoluzione meno un byte).

La prima linea del nostro programma sarà così strutturata:

```
0 POKE 56,8191/256: POKE 55,8191-
(PEEK(56)*256): CLR
1 REM * qui comincia il programma
.....(eccetera)...............
```

Anche qui la costante 8191 rappresenta l'indirizzo della locazione posta come fine dell'area utilizzabile dal Basic e pertanto deve essere modificata a seconda delle esigenze.

Ricapitolando ciò che abbiamo appreso in questo paragrafo, potremo affermare che:

computer service

VENDITA PER CORRISPONDENZA

ACCESSORI PER COMPUTER COMMODORE

GRUPPO CONTINUITÀ

Fornito senza le 12 batterie a stilo ricaricabili. Consente il funzionamento del Vostro computer Commodore C64 o VIC 20 in assenza di corrente. Durata di funzionamento 30 minuti. Ricarica tramite alimentatore Commodore.

KIT ALLINEAMENTO TESTINA

Composto dal cacciavite, nastro di controllo e strumento di taratura con monitor audio permette il perfetto allineamento dei registratori digitali anche con nastri commerciali.

VELOCIZZATORE DI CARICAMENTO FLOPPY

Cartridge con un insieme di utility residenti su ros per velocizzare il drive nel Commodore 64.

INTERFACCIA RADIO

Indispensabile per registrare con registratore Commodore modello "C2N" i programmi speciali per computer trasmessi dalle emittenti radio.

CUFFIA PER COMMODORE C 64

Leggerissima permette l'ascolto personale del computer evitando di disturbare durante i giochi.

COPIATORE PROGRAMMI

Dispositivo hardware per effettuare copie di nastri protetti o turbo utilizzando due registratori Commodore o compatibili.

DUPLICATORE CASSETTE

Indispensabile per realizzare delle copie, con un registratore normale, di un nastro protetto o con caricamento turbo

Articolo	Art.	L.
Bus quadrislot	Art. CD 100	L. 55.000
Interfaccia cassette	Art. CD 101	L. 30.000
Duplicatore cassette	**Art. CD 102**	**L. 30.000**
Copiatore programmi	**Art. CD 103**	**L. 30.000**
Interfaccia radio	**Art. CD 104**	**L. 30.000**
Kit allineamento testina	**Art. CD 105**	**L. 47.000**
Alimentatore per C64 e VIC 20	Art. CD 106	L. 45.000
Gruppo continuità (fornito senza le 12 batterie a stilo ricaricabili)	**Art. CD 107**	**L. 66.000**
Pacco batterie (12 stilo 1,2 Volt ricaricabili	Art. CD 117	L. 52.000
Commutatore antenna TV/computer	Art. CD 108	L. 9.500
Tasto reset	Art. CD 109	L. 5.500
Interfaccia Centronics	Art. CD 112	L. 104.000
Espansione di memoria per C 16	Art. CD 114	L. 158.000
Velocizzatore di caricamento flop.	**Art. CD 115**	**L. 49.000**
Espansione di memoria per VIC 20 16K	Art. CD 116	L. 112.000
Modulatore Executive	Art. CD 120	L. 72.000
Penna ottica grafica	Art. CD 121	L. 45.000
Tavoletta grafica	Art. CD 130	L. 238.000
Multipresa con filtro - 2 prese	Art. CD 140	L. 41.000
Cuffia per Commodore C 64	**Art. CD 150**	**L. 19.000**
Stabilizzatore elettronico di tensione 500 W	Art. CD 160	L. 430.000
Gruppo di cintinuità 60 W	Art. CD 170	L. 400.000
Gruppo di continuità 200 V	Art. CD 180	L. 802.000
Inverter 12 Volt cc. 220 Volt ca. 100 Watt	Art. CD 190	L. 297.000
Cavo alimentazione	Art. CD 200	L. 4.600
Cavo drive o stampante Commodore	Art. CD 205	L. 8.500
Prolunga per Joystick - mt. 3	Art. CD 210	L. 25.000
Prolunga per cavo TV - mt. 3	Art. CD 215	L. 12.500
Cavo audio - mt. 6	Art. CD 220	L. 15.500
Adattatore Joistick (Atari e C64 al C 16)	Art. CD 225	L. 10.500
Adattatore registratore per C 16	Art. CD 226	L. 19.500
Mascherina antiriflesso 12"	Art. CD 300	L. 35.000
Nastro inchiostrato per Tally - mt. 80	Art. CD 610	L. 16.500
Nastro inchiostrato per Tally - mt. 180	Art. CD 611	L. 16.500
Nastro inchiostrato per Tally 1000 e Honeywell	Art. CD 612	L. 9.500
Nastro inchiostrato per Commodore MRS 801	Art. CD 614	L. 13.000
Nastro inchiostrato per Commodore MPS 802	Art. CD 616	L. 18.000
Nastro inchiostrato per Commodore MPS 803	Art. CD 618	L. 19.500
Mause per Commodore C 64	Art. CD 860	L. 240.000
Pacco carta lettura facilitata 24" x 11" modulo da 500 fogli con bordi a strappo	Art. CD 630	L. 13.500
Supporto stampante porta carta in plexiglass "fume" - normale	Art. CD 660	L. 59.000
Supporto stampante porta carta in plexiglass "fume" - rinforzato	Art. CD 670	L. 80.000
Floppy disk 5" singola faccia doppia densità "ODP" - conf. 10 pezzi	Art. CD 700	L. 40.000
Floppy disk 5" singola faccia doppia densità "CBS" - conf. 10 pezzi	Art. CD 702	L. 38.000
Floppy disk 5" singola faccia doppia densità "VERBATIM" - conf. 10 pezzi	Art. CD 704	L. 42.000
Floppy disk 5" singola faccia doppia densità "DYSAN" - conf. 10 pezzi	Art. CD 706	L. 68.000
Nastri magnetici C 10 digitali - conf. 10 pezzi	Art. CD 712	L. 20.000
Nastri magnetici C 15 digitali	Art. CD 714	L. 21.000
Copritastiera in plexiglass per C64 - C16 e VIC 20	Art. CD 750	L. 16.000
Copritastiera in stoffa per C64 - C16 e VIC 20	Art. CD 760	L. 10.500
Vaschetta portafloppy in plexiglass per 40 dischi con chiave	Art. CD 770	L. 30.000
Vaschetta portafloppy in plexiglass per 90 dischi con chiave	Art. CD 780	L. 37.000
Kit pulizia testine registratore	Art. CD 815	L. 13.500
Kit pulizia disk drive	Art. CD 820	L. 26.000
Kit pulizia tastiera	Art. CD 830	L. 16.500
Foratore disk in plastica (per utilizzare la seconda faccia dei dischi)	Art. CD 840	L. 10.000
Foratore disk in metallo "tako"	Art. CD 849	L. 14.000
Joystick Spectravideo II	Art. CD 850	L. 27.000
Joystick a Microswitch	Art. CD 851	L. 52.500
Joystick senza fili con unità ricevente (funziona a batteria)	Art. CD 852	L. 98.000
Joystick per Commodore 16 (originale)	Art. CD 130	L. 29.500

TUTTI I PREZZI SONO COMPRENSIVI DI IVA
NON SI ACCETTANO ORDINI INFERIORI A L. 30.000
CONTRIBUTO FISSO SPESE DI SPEDIZIONE L. 5000

SI ACCETTANO ANCHE ORDINI TELEFONICI
AI NUMERI 0522/661647-661471

BUONO DI ORDINAZIONE

NOME - COGNOME

INDIRIZZO

N.

C.A.P. CITTÀ PROVINCIA

VOGLIATE INVIARMI IN CONTRASSEGNO

N.	Art.	L.
N.	Art.	L.
N.	Art.	L.
SPESE SPEDIZIONE		L. 5.000
PAGHERÒ AL POSTINO		L.

COMPUTER SERVICE VIA A. MANZONI, 49 - 42017 NOVELLARA (RE) - TEL. (0522) 661647

- *Non si può utilizzare l'area 0-1023.*
- *Le locazioni della pagina grafica non devono coincidere, nemmeno in parte, con l'area riservata ai programmi Basic.*

Banco n. 2

Come si può notare dalla mappa di memoria della tabella N.3, questo banco non crea particolari problemi in quanto non vi sono aree destinate ad un uso particolare:

Tabella N.3

Zona RAM del terzo banco	0 16383
Area RAM riservata ai programmi ed alle variabili Basic	16384 32767
Zona RAM del primo banco	32768 49151

L'unico fastidio di una pagina grafica situata in queste locazioni, consiste nella divisione a metà effettuata nella memoria adibita al Basic.

In questo modo, per avere i migliori risultati, bisogna calcolare esattamente la lunghezza del programma e poi decidere se modificare l'inizio o la fine del Basic.

Consigliamo, per quanto possibile, di optare per la seconda soluzione per motivi di praticità, anche se l'ideale sarebbe non utilizzare questo banco di memoria.

Le operazioni da compiere per modificare i puntatori 43-44 e 55-56 sono le stesse descritte nel paragrafo precedente.

Banco N. 1

I veri problemi nascono proprio quando si utilizza questo banco come pure il N.0. Ci si può rendere conto di ciò osservando la mappa di memoria della tabella N.4.

Tabella N.4

Zona RAM dei primi due banchi	0 32767
Area RAM riservata ai programmi e alle variabili Basic	32768 40959
ROM interprete Basic oppure RAM libera	40960 49151
Zona RAM del quarto blocco	49152 65535

Tutto è come prima finchè non si oltrepassa la locazione 40959; non appena si tenta di manipolare l'area in cui normalmente è situato l'interprete Basic su ROM, ci si rende conto che "qualcosa" non va per il verso giusto.

Per capire bene ciò che fra poco diremo, bisogna sapere, innanzitutto, che il Commodore 64 ha ben più di 64K byte di memoria: infatti vi sono anche 16K byte di ROM che supportano il KERNAL (cioè il Sistema Operativo) ed il Basic (linguaggio).

Siccome il microprocessore 6510 non può riconoscere che 64K byte di memoria per volta, per farlo accedere anche alla memoria RAM "sottostante" alle ROM, i progettisti della Commodore hanno dotato il microprocessore di un chip supplementare, chiamato PLA (Programmable Logic Array), che interpreta gli indirizzi dati dal suo "capo" abilitando la RAM oppure, a seconda dei casi, la ROM.

Per abilitare la RAM nella zona che va da 40960 a 49151, si resetta (si pone a zero) il bit zero (0) della locazione 1 con l'istruzione:

```
POKE1, PEEK(1)AND 254
```

Tutti coloro che digiteranno però questa istruzione rimarranno profondamente delusi in seguito al "blocco" del C/64.

Per riportare tutto in ordine, è sufficiente (!) spegnere e riaccendere il computer.

L'istruzione di prima riporta la RAM a galla, ma, inevitabilmente, "affonda" la ROM del Basic che, ovviamente, non può espletare più le funzioni che le competono.

L'unico modo per manipolare queste locazioni RAM è quello di ricorrere al linguaggio macchina.

Viceversa, il Vic II accede sempre a locazioni RAM della memoria (tranne quando si serve della ROM generatrice dei caratteri) anche se il microprocessore sta, nello stesso momento, accedendo a locazioni ROM.

Questo è stato un innegabile vantaggio per coloro che hanno creato tool grafici come il Simon's Basic: infatti quest'ultimo opera su una pagina grafica in alta risoluzione che risiede proprio sotto il KERNAL.

Così facendo non si sottrae memoria al Basic e non si hanno, di conseguenza, problemi di sorta con i puntatori d'inizio e fine programma.

Come necessaria contropartita di tale efficienza operativa, una pagina grafica situata sotto la ROM è notevolmente difficile da gestire.

Concludendo il presente paragrafo, possiamo consigliare di usare solo la memoria situata fino a 40959 in modo da evitare complicazioni non indifferenti.

Banco n. 0

Questa del banco N.0 è la mappa di memoria più complicata di tutte in quanto prevede, per una certa lunghezza, la presenza contemporanea di ben tre diversi dispositivi di memorizzazione.

Tabella 5	
Zona dei banchi 3,2,1	0 49151
Area RAM libera	49152 53247
I/O (Vic, SID, CIA) e RAM colore o ROM dei caratteri o RAM libera	53248 57343
ROM KERNAL o RAM libera	57344 65535

L'area di 4K byte che va da 49152 a 53247 è molto comoda da usare per le pagine grafiche in modo testo sulle quali potranno comparire caratteri ridefiniti.

Nell'articolo di CCC N.30 è stato appunto presentato un mini gioco che situava la pagina testo a partire da 50176 e la nuova mappa dei caratteri da 51200. In questo modo si poteva ancora disporre di 1024 byte liberi (49152-59175) da adibire alla memorizzazione di eventuali sprites o routines in linguaggio macchina, senza disturbare minimamente il Basic.

A differenza di quest'ultima, la zona di memoria compresa da 53248 a 57343 è la più difficile da gestire in quanto proprio in queste locazioni vi sono i registri del Vic II che, normalmente presenti, vengono esclusi allorquando si abilita la ROM generatrice dei caratteri o la RAM libera sottostante.

Per solo dovere di cronaca diciamo che, per abilitare la RAM sottostante, dovremo mettere a 0 i bit 0 e' 1 della locazione 1 con l'istruzione:

POKE 1,PEEK(1) AND 252

Facciamo osservare che questa operazione abilita tutta la RAM disponibile, anche quella sotto il KERNAL e sotto il Basic, facendo del C/64 un sistema composto di sola memoria RAM.

Volendo, nonostante tutto, utilizzare le locazioni da 53248 a 57343, dovremo innanzitutto adoperare un programma in L.M., quindi disabilitare gli interrupts con l'istruzione Assembler SEI e, infine, portare a galla la RAM manovrando la locazione 1 come descritto sopra.

L'utilizzazione dell'area 57348-65535 è notevolmente comoda nel caso si voglia adoperare una pagina grafica in alta risoluzione senza sottrarre memoria al Basic.

L'unico inconveniente è quello, solito, di dover ricorrere al linguaggio macchina in quanto l'abilitazione della RAM porta alla automatica esclusione non solo del KERNAL, ma anche del Basic.

Ancora una volta, per concretizzare quanto detto, dobbiamo modificare il valore della locazione 1 e, più precisamente, del bit 1 che dovremo porre a zero con l'istruzione:

POKE 1,PEEK(1) AND 253

L'istruzione Basic è riportata solo per chiarezza; è evidente che una volta eseguita, il sistema si blocca non trovando più il KERNAL ed il Basic. Volendo usarla concretamente è necessario "tradurla" in L.M.

Il programma-tabella

Per facilitare al massimo le complicate operazioni sui registri del Vic II, abbiamo approntato un programma in Basic che, con una serie di Input, fornisce gli argomenti da assegnare alle istruzioni POKE in modo da raggiungere il risultato voluto.

Sono contemplati tutti i modi grafici possibili, in alta e bassa risoluzione, in modo normale, in multicolor e in extended background, non trascurando i caratteri riprogrammabili.

Il programma è stato realizzato in modo da guidare l'utente attraverso le sue scelte e fornisce, come Output, le POKE necessarie man mano che si procede nell'individuazione dell'obiettivo prefissato.

Onde evitare il crash del sistema, il programma si limita a visualizzare le operazioni da compiere per la creazione della pagina grafica o dei caratteri ridefiniti: sarà compito dell'utente la loro gestione, presupponendo, pertanto, la conoscenza del meccanismo di base che regola i vari modi grafici (che si può raggiungere con la lettura dei due articoli precedenti - fine della pubblicità).

Le "features" del programma sono completate da una serie di messaggi che avvertono l'utente degli eventuali problemi riscontrabili nella scelta di una determinata opzione grafica.

Conclusioni

Come affermato all'inizio di questo articolo, nella redazione del testo è stato utilizzato un liguaggio abbordabile dalla maggior parte degli utenti. E' stato anche usato uno schema di trattazione che, pur essendo generico, cerca di allontanarsi dalla superficialità specificando, caso per caso, l'atteggiamento da assumere per ogni singolo problema dandone, per molti, anche la soluzione pratica.

Naturalmente solo l'utilizzazione del linguaggio macchina porta ai risulati più interessanti nel campo della grafica ma, per evidenti problemi di spazio e, soprattutto, di comprensione, l'argomento è stato solo accennato.

Di ciò ne potremo parlare più approfonditamente in seguito nel caso che vi fosse un manifesto interesse da parte dei lettori.

```
100 REM GRAPHIC MANAGER
110 REM PER COMMODORE 64
120 REM BY PASQUALE D'ANDRETI
130 REM RIARDO (CE)
140 :
150 X$(1)="ALTA RISOLUZIONE"
160 X$(2)="MODO TESTO"
170 X$(3)="CARATTERI RIDEFINIBI
    LI"
180 :
190 Y$(1)="NORMALE"
200 Y$(2)="MULTICOLOR"
210 Y$(3)="EXT.BACKGROUND"
220 :
230 Z$(0)="49152-65535"
240 Z$(1)="32768-49151"
250 Z$(2)="16384-32767"
260 Z$(3)="     0-16383"
270 :
280 POKE 53280,0:POKE 53281,0:P
    RINTCHR$(147)CHR$(8)
290 PRINTCHR$(18)"GRAPHIC MANAG
    ER - BY PASQUALE D'ANDRETI"
    :PRINT
300 PRINT"VUOI UTILIZZARE:":PRI
    NT
310 PRINT"1  "X$(1)
320 PRINT"2  "X$(2)
330 PRINT"3  "X$(3):PRINT
340 LI=1:LS=3:GOSUB 1080:MG=C:X
    =C
350 ON MGGOTO 360,480,590
360 PRINTCHR$(147)
370 PRINT"QUALE MODO DI VISUALI
    ZZAZIONE SCEGLI ?":PRINT
380 PRINT"1  "Y$(1)
390 PRINT"2  "Y$(2):PRINT
400 LI=1:LS=2:GOSUB 1080
410 CC=C:Y=C:GOSUB 1000
420 PRINTCHR$(147)
430 PRINT"DOVE VUOI POSIZIONARE
     LA PAG. GRAFICA?"
440 I=(3-BN)*16384:F=I+16383:IN
    =8192:GOSUB 950:PG=C
450 PRINT"DOVE VUOI POSIZIONARE
     LA PAG. COLORE?"
460 I=I-(8192*(PG=0)):F=I+8191:
    IN=1024:GOSUB 950:PC=C
470 GOTO 640
480 PRINTCHR$(147)
490 PRINT"QUALE MODO DI VISUALI
    ZZAZIONE SCEGLI?":PRINT
500 PRINT"1  "Y$(1)
510 PRINT"2  "Y$(2)
520 PRINT"3  "Y$(3):PRINT
530 LI=1:LS=3:GOSUB 1080
540 CC=C:Y=C:GOSUB 1000
550 PRINTCHR$(147)
560 PRINT"DOVE VUOI POSIZIONARE
     LA PAGINA TESTO?"
570 I=16384*(3-BN):F=I+16383:IN
    =1024:GOSUB 950
580 PT=C:GOTO 640
590 GOSUB 1000
600 PRINTCHR$(147)
610 PRINT"DOVE VUOI POSIZIONARE
     LA NUOVA MAPPA ?"
620 I=16384*(3-BN):F=I+16383:IN
    =2048:GOSUB 950
630 NM=C
640 PRINTX$(X)
650 IF X<3 THEN PRINT"MODO DI V
    ISUALIZZAZIONE:"Y$(Y)
660 PRINT"BANCO DI MEMORIA:"Z$(
    Z)
670 PRINTCHR$(18)"PREMI UN TAST
    O PER CONTINUARE"
680 POKE 198,0:WAIT 198,1
690 PRINTCHR$(147)CHR$(18)"OSSE
    RVAZIONI SUL BANCO DI MEMOR
    IA SCELTO":PRINTX$(X)
700 ON BN+1GOSUB 1120,1200,1260
    ,1330
710 PRINTCHR$(18)"PREMI UN TAST
    O PER CONTINUARE"
720 POKE 198,0:WAIT 198,1
730 PRINTCHR$(147)"ECCO LE ISTR
    UZIONI DA DIGITARE:":PRINT
740 PRINT"POKE 56576,(PEEK(5657
    6) AND 252) +"BN
750 ON MGGOSUB 820,880,930
760 PRINT:PRINT"QUESTE ISTRUZIO
    NI AVRANNO UN RISCONTRO"
770 PRINT"POSITIVO SOLO SE SONO
     STATE OSSERVATE LE";
780 PRINT"PRECAUZIONI DESCRITTE
     NELL' ARTICOLO."
790 PRINTCHR$(18)"VUOI RIPETERE
```

```
     I CALCOLI (S/N)?
 800 POKE 198,0:WAIT 198,1:GET A
     $:IF A$="S" THEN  RUN
 810 END
 820 PRINT"POKE 53265,PEEK(53265
     ) OR 32
 830 IF CC=2 THEN PRINT"POKE 532
     70,PEEK(53270) OR 16
 840 PRINT"POKE 53272,PEEK(53272
     ) OR ";:IF PG=0 THEN PRINT"
     1":GOTO 860
 850 PRINT"8
 860 PRINT"POKE 53272,(PEEK(5327
     2) AND 15) OR"PC*16
 870 RETURN
 880 IF CC=2 THEN PRINT"POKE 532
     70,PEEK(53270) OR 16
 890 IF CC=3 THEN PRINT"POKE 532
     65,PEEK(53265) OR 64
 900 PRINT"POKE 53272,(PEEK(5327
     2) AND 15) OR"PT*16
 910 PRINT"POKE 648,"((3-BN)*163
     84+PT*1024)/256
 920 RETURN
 930 PRINT"POKE 53272,(PEEK(5327
     2) AND 241) OR "2*NM
 940 RETURN
 950 W=0
 960 PRINT"   [RVS]"W"[RVOFF]"I+(
     IN*W)CHR$(13)"[UP]"
 970 W=W+1:IF I+(IN*W)<=F THEN 9
     60
 980 LI=0:LS=W-1:GOSUB 1080
 990 RETURN
1000 PRINTCHR$(147)
1010 PRINT"QUALE BANCO DI MEMORI
     A VUOI USARE?"
1020 PRINT"0- "Z$(0)
1030 PRINT"1- "Z$(1)
1040 PRINT"2- "Z$(2)
1050 PRINT"3- "Z$(3)
1060 LI=0:LS=3:GOSUB 1080:BN=C:Z
     =C
1070 RETURN
1080 INPUT "SCELTA:[RVS]";C$
1090 C=INT(ABS(VAL(C$)))
1100 IF C<LI OR C>LS THEN 1080
1110 RETURN
1120 PRINT"LE LOCAZIONI DA 49152
      A 53247 SI POSSONO";
1130 PRINT"USARE LIBERAMENTE ANC
     HE DA BASIC, MENTRE";
1140 PRINT"QUELLE DA 53248 A 655
     35 HANNO BISOGNO DI";
1150 PRINT"UNA GESTIONE DA LINGU
     AGGIO MACCHINA."
1160 PRINT"SI RAMMENTI CHE GLI I
     NDIRIZZI DA 53248 A";
1170 PRINT"57343 CONTENGONO ANCH
     E LA MAPPA DEI  CA-";
1180 PRINT"RATTERI IN ROM E I RE
     GISTRI DEL VIC II."
1190 RETURN
1200 PRINT"LE LOCAZIONI DA 32768
      A 40959 SI POSSONO";
1210 PRINT"ANCHE USARE DA BASIC,
      ABBASSANDO  OPPOR-";
1220 PRINT"TUNAMENTE IL PUNTATOR
     E 55-56."
1230 PRINT"LE RESTANTI LOCAZIONI
      (40960-49151) SONO";
1240 PRINT"UTILIZZABILI SOLO DA
     LINGUAGGIO MACCHINA"
1250 RETURN
1260 PRINT"QUESTO BANCO E' INTER
     AMENTE UTILIZZABILE";
1270 PRINT"DA BASIC."
1280 PRINT"PRIMA DI CREARE LA  P
     AGINA  GRAFICA, E'"
1290 PRINT"NECESSARIO ABBASSARE
     O ALZARE I PUNTATO-";
1300 PRINT"D'INIZIO E FINE BASIC
      LOCATI IN 43-44 E"
1310 PRINT"55-56."
1320 RETURN
1330 PRINT"IN QUESTO BANCO GLI I
     NDIRIZZI CHE VANNO"
1340 PRINT"DA 0 A 1023 NON POSSO
     NO ESSERE USATI PER";
1350 PRINT"LA MEMORIZZAZIONE DI
     PAGINE GRAFICHE."
1360 PRINT"PRIMA DI USARE IL PRO
     GRAMMA CHE GESTISCE";
1370 PRINT"LA GRAFICA, BISOGNA A
     DATTARE I PUNTATORI";
1380 PRINT"D'INIZIO E FINE BASIC
     ."
1390 RETURN
```

COMPUTER COMMODORE e DRIVE 1541

Cambiamo il numero di device

Come applicare un microinterruttore per cambiare numero alla periferica.

di Michele Maggi

Il drive 1541 della Commodore, contrariamente al nuovo 1571, non ha la possibilità di modificare il numero di device tramite microinterruttore.

Esiste però (a parte la procedura software di cui parleremo più avanti) una possibilità hardware per effettuare la modifica.

Questa consiste nel tagliare una pista del circuito stampato modificando però "per sempre " il numero di riferimento del proprio drive.

Il suddetto metodo ha chiaramente un difetto: una volta modificato il numero di device, il drive rimarrà fisso a 9 fino al prossimo "intervento ".

Chiaramente tale procedura è consigliata solo a persone in possesso di più drive, ma può tornare utile anche a chi, di drive, ne possiede uno solo.

Va da sè che la modifica hardware, alla lunga, può creare problemi soprattutto nel caricamento di programmi composti da più file in quanto, una volta partito il programma principale, si verificherà un sicuro crash del sistema perchè i restanti file verranno cercati sul drive 8 il quale, non essendo presente....

Vediamo come si può intervenire inserendo un microinterruttore che permetta di selezionare il numero di device fra 8 e 9.

Gli "ingredienti " necessari sono pochissimi e il loro costo addirittura irrisorio.

Si tratta di un microinterruttore e di mezzo metro di filo elettrico.

Chi non se ne intende di hardware, ed sè già spaventato all'idea di aprire il drive, si tranquillizzi: questa modifica è di notevole semplicità tanto che io stesso (digiuno di hardware) sono riuscito a realizzarlo senza problemi.

Procedimento

Per prima cosa apriamo il drive e individuiamo i due contatti, perfettamente identici, che sovraintendono al numero di device (figura 1).

Successivamente, servendosi di un sottile cacciavite (ma è meglio prendere una taglierina) bisogna tagliare la pista del contatto più in basso, quello, tanto per intenderci, più vicino alla porta del drive (figura 2).

A questo punto il nostro drive ha, a tutti gli effetti, 9 come numero di device.

Il terzo "step " consiste nel saldare il filo elettrico a ciascuna delle metà del

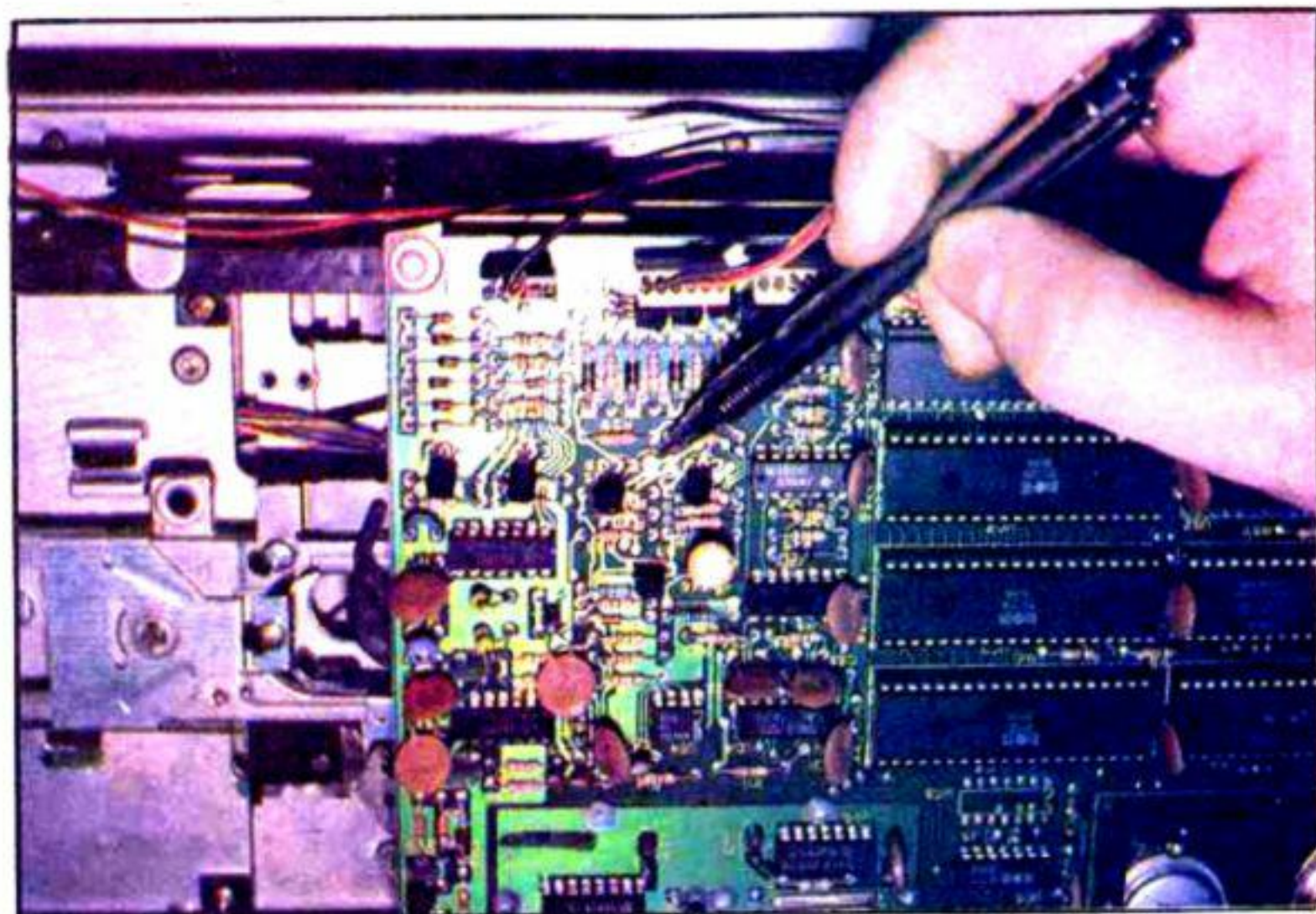



Foto 1 La pista del contatto è stata tagliata.

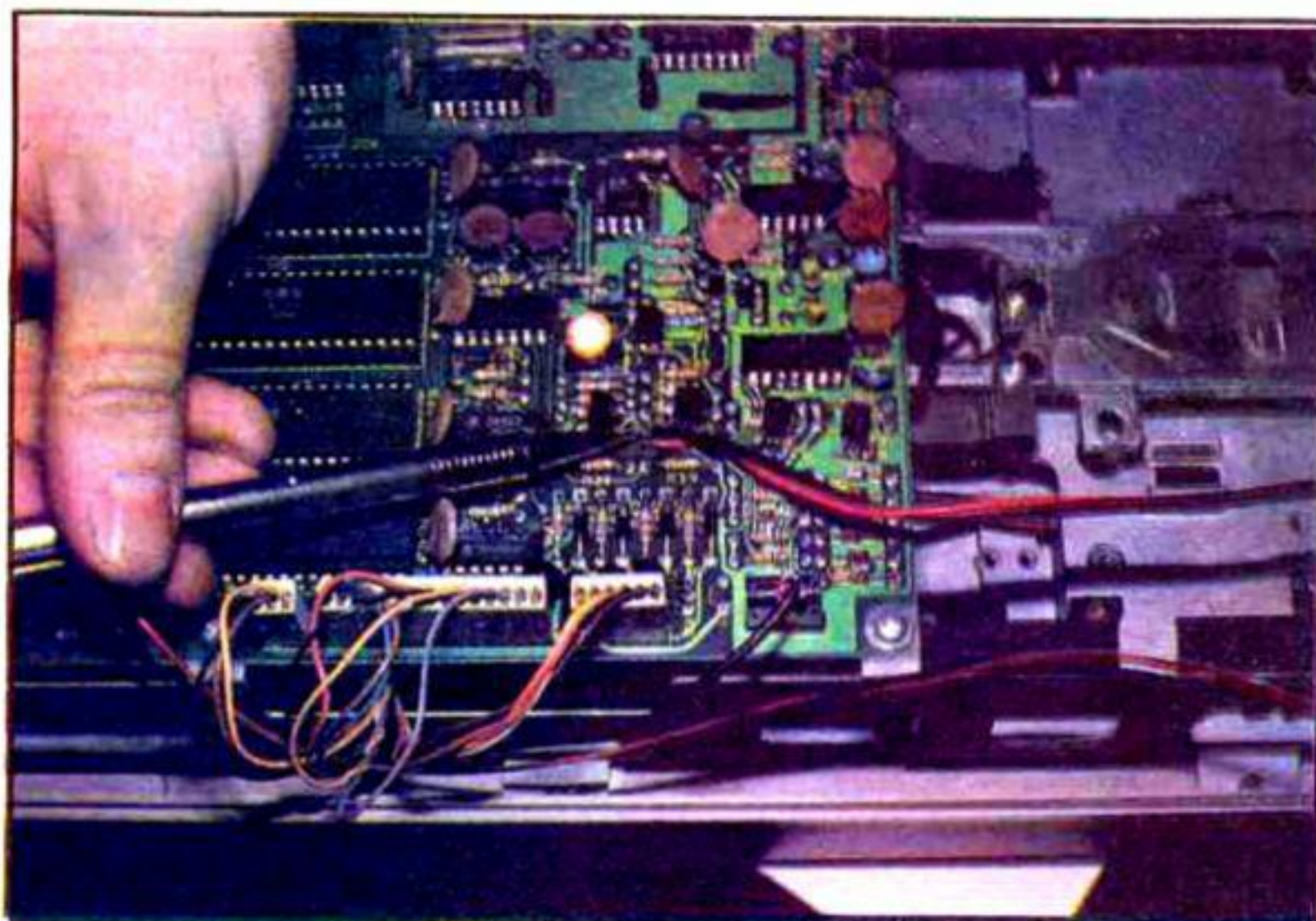



Foto 2 La saldatura del filo elettrico al contatto.

contatto collegando le estremità al microinterruttore (figura 3).

A questo punto, se avete fedelmente seguito le istruzioni, il vostro drive avrà 8 oppure 9 come numero di device in funzione dell'apertura o chiusura dell'interruttore.

L'ultima operazione da compiere consiste nel praticare un foro in un punto "spazioso" della calotta del drive e inserirvi l'interruttore.

Chiudete il drive e... il gioco è fatto.

Chi avesse comunque necessità di cambiare numero di device da 8 a 9 in modo "temporaneo" potrà digitare il brevissimo listato di queste pagine ed utilizzarlo ogni volta che ve ne fosse bisogno.

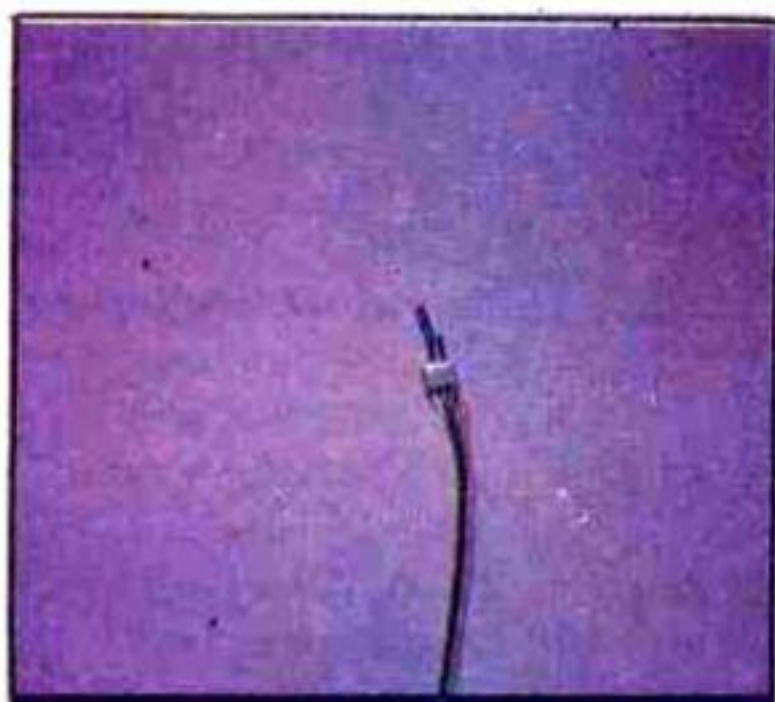



Foto 3 Saldatura del filo al microinterruttore.

```
10 INPUT "N. VECCHIO
   DEVICE";DV

15 IF DV<8 OR DV>15 T
   HEN 10

20 INPUT "N. NUOVO DE
   VICE";DN

25 IF DN<8 OR DN>15 T
   HEN 20

30 OPEN 15,DV,15

40 PRINT#15,"M-R"CHR$
   (255)CHR$(255)

50 GET #15,A$:A=ASC(A
   $+CHR$(0)):TY=0:HI
   =0

60 IF A=242 OR A=213
   THEN TY=12:GOTO 10
   0:REM  DRIVE 8050
   OR 4040

70 IF A=254 THEN TY=1
   19:GOTO 100:REM  D
   RIVE 2031 OR 1541

80 IF A=226 THEN TY=5
   0:GOTO 100:REM  DR
   IVE 3040

90 PRINT"DRIVE NON CO
   MPATIBILE":CLOSE 1
   5:END

100 PRINT#15,"M-W"CHR$
    (TY)CHR$(HI)CHR$(2
    )CHR$(32+DN)CHR$(6
    4+DN)

110 CLOSE 15
```

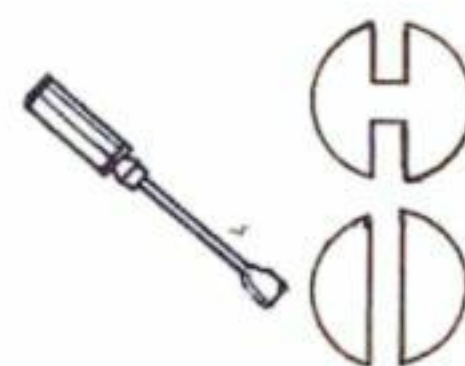



Foto 4 Alloggiamento del microinterruttore.

"Micro" Pascal per computer Commodore

E' possibile scrivere, in Basic, un altro linguaggio evoluto? L'ultimo libro della Systems Editoriale ci tenta. E con risultati niente male.

Il Pascal è un linguaggio evoluto che ricorre ad una logica del tutto nuova e particolare per risolvere problemi.

La decisione di stendere un programma in grado di rispettare la sintassi del noto linguaggio è stata portata a termine dall'ingegner Clizio Merli (nostro valido collaboratore, specializzato da anni in telematica) su idea di Paolo Caprioglio nel corso di laurea in Ingegneria presso il Politecnico dell'università di Torino.

Il programma "gira" su qualsiasi sistema CP/M e, per ciò che ci riguarda, su C/64, C/128 e anche (lo abbiamo scoperto poi...) su qualsiasi altro computer dotato di cospicua memoria (come il Plus/4, il Vic 20 e il C/16 espansi almeno a 38 Kbyte RAM).

In appendice, infatti, sono riportate due versioni dello stesso programma: la prima, per computer CP/M (Basic Standard Microsoft), occupa 11 pagine di listato; la seconda, per C/64, ha richiesto poco più di otto pagine.

A chi può servire

Come, del resto, già specificato nello stesso volumetto, il programma non deve esser visto come strumento per "lavorare" in Pascal, benchè ne consenta la struttura. L'opera va considerata, piuttosto, sotto l'aspetto didattico sia perchè è possibile rendersi conto di come si può progettare un linguaggio (partendo da ipotesi di strutture), sia perchè, dato che è scritto interamente in Basic, è possibile modificarlo a piacere, sia perchè consente di scrivere un programma secondo la corretta sintassi del Pascal. Anche se considerato soltanto da quest'ultimo punto di vista, il libro risulta un valido manuale di istruzioni indispensabile per coloro che, avendo occasione di lavorare con altre versioni di Pascal, non hanno ben chiara la sua logica e la sintassi adoparata.

Un libro molto chiaro

E' difficile convincere un lettore della chiarezza espositiva di un testo recensito. Consci di tale fatto, ci limitiamo a pubblicare, integralmente, una parte del libro stesso, dedicato alla "Programmazione strutturata", le cui argomentazioni sono valide indipendentemente dal linguaggio o procedure adoperati in qualsiasi computer.

Come procurarsi libro e programma

Il volume è in vendita a L.7000 in tutte le edicole e contiene, al suo interno, una cedola per l'acquisto a prezzo privilegiato (L.12500, compresa la spedizione) del nastro contenente il programma per C/64.

I lettori che lo desiderano possono ordinare sia il libro che il nastro inviando un totale di L.19500 (sotto forma di assegno bancario, vaglia o C/C postale N.37952207) intestato a:
Systems Editoriale - Viale Famagosta 75 c.a.p. 20142 Milano - specificando con la massima chiarezza nome, cognome, indirizzo e titolo del volume richiesto (≈ Pascal per C/64 e nastro).

Non ci è possibile inviare materiale in contrassegno.

La programmazione strutturata

Parlare di "programmazione strutturata" è diventato un fatto di moda, che viene da molti dato ormai per scontato. Ma spesso si dà al termine un significato errato, legandolo a questo o quel particolare linguaggio od elaboratore.

In realtà la *programmazione strutturata* è una particolare tecnica di analisi e di programmazione abbastanza indipendente dal linguaggio o dall'elaboratore utilizzati. D'altro canto esistono specifici linguaggi di programmazione per i quali la strutturazione è d'obbligo per un corretto sviluppo dei programmi.

L'utilizzo della programmazione strutturata può risultare, a chi lo affronta per la prima volta, abbastanza ridondante e noioso. Le aree dati devono essere definite a priori, secondo regole precise che ammettono poche eccezioni, specificando con accuratezza le varie strutture (i tipi), i valori ammessi, le operazioni consentite e così via. La scrittura delle istruzioni può essere effettuata solo dopo aver analizzato tutti gli aspetti del problema, con una particolare enfasi sul corretto controllo del flusso di esecuzione. In alcuni casi, utilizzando linguaggi fortemente orientati ai tipi dei dati, la corretta definizione ed utilizzo dei sottoprogrammi può richiedere lunghe ore di controlli e verifiche tra una compilazione e l'altra, in quanto ogni minima incongruenza viene rilevata e segnalata dal compilatore. Un classico esempio di questa situazione è il linguaggio Pascal, la cui precisione viene spesso giudicata ai limiti dell'assurdo da chi mal sopporta la disciplina ed il rigido formalismo.

Apparentemente non c'è nulla di così eccezionale in questo modo di procedere, anzi. Tutti i testi di analisi e programmazione degni di tale nome descrivono il processo di realizzazione di un program-

ma esattamente in questi termini. Ma chi è abituato a scrivere in Basic, od in assembler e non ha mai affrontato la realizzazione di programmi di grosse dimensioni, può rimanere sconcertato da questa pignoleria tipicamente prussiana, che sembra togliere quasi ogni spazio alla fantasia ed all'estro creativo.

Proviamo però ad immaginare la realizzazione di un insieme di programmi complessi, costituiti da migliaia di istruzioni, quale può essere un pacchetto di gestione integrata per una media azienda, o un pacchetto di analisi statistica. Chi affronta problemi di questa portata come un novello Beethoven dell'informatica "creando" i programmi direttamente alla tastiera con un'analisi preliminare ridotta ai minimi termini, è sicuramente votato al fallimento.

Ma anche nella remota ipotesi di una fortuna sfacciata, che permetta di ottenere una prima versione funzionante dei suoi programmi, alla prima modifica corre il rischio certo di vedere cadere in un soffio tutto il suo complesso castello di carte. E non conterà più i giorni e le notti passati nel disperato tentativo di rimettere in funzione la sua macchina "quasi perfetta".

In effetti la concezione di un algoritmo di media complessità, utilizzando direttamente le istruzioni elementari di un qualunque linguaggio di programmazione, è un approccio che scoraggia in breve tempo anche il più paziente e testardo dei programmatori. L'uomo infatti, è abituato a ragionare analizzando di volta in volta un numero limitato di elementi, senza mai (o quasi mai) operare contemporaneamente su tutti gli aspetti (grandi e piccoli) di un prolema.

Un primo importante assioma della programmazione strutturata riguarda appunto la scomposizione del problema in un certo numero di blocchi di "alto livello". Ogni blocco viene visto come una potente istruzione di una macchina virtuale, sulla cui struttura si fanno inizialmente solo delle ipotesi. Quando il problema è risolto utilizzando questi blocchi funzionali, si inizia ad esaminare più in dettaglio ogni singolo blocco, ripetendo eventualmente il processo ed ipotizzando ogni volta l'esistenza di macchine virtuali sempre meno potenti. Il lavoro termina quando tutti i blocchi funzionali dei vari livelli sono stati definiti in base alle caratteristiche funzionali del linguaggio di programmazione prescelto.

Questo approccio viene detto di tipo "top-down" (dall'alto verso il basso). Un successivo riesame in senso inverso ("bottom-up", da basso in alto) permette di verificare la correttezza dell'analisi effettuata ed eventualmente di correggerla e migliorarla.

Ogni "blocco" così definito viene detto *sottoprogramma*, in quanto rappresenta una parte più piccola del programma complessivo, che può essere richiamata in diversi punti del programma, senza dover riscrivere ogni volta la stessa sequenza di istruzioni. Alla fine il programma riulta spezzato in varie unità, ciascuna delle quali compie un'elaborazione autonoma che risolve una parte del problema. I vari sottoprogrammi, opportunamente connessi, costituiscono il programma completo.

Un altro e più importante aspetto della programmazione strutturata è relativo alla definizione dei dati, le informazioni che devono essere trattate del programma. La definizione di un blocco funzionale (un'istruzione di una macchina virtuale più potente di quella realizzata dal linguaggio di programmazione adottato) presuppone la definizione del tipo di informazioni trattate dal blocco e quindi della loro "struttura". L'aggettivo "strutturato" è infatti rivolto più ai dati che non alle specifiche istruzioni.

Il primo passo dell'analisi di un programma deve essere rivolto alla esatta individuazione dei tipi di informazioni trattate, alle loro caratteristiche, alle modifiche cui devono essere assoggettate durante l'elaborazione. In una parola, alla loro "struttura".

Teo Rusconi ha appena sfatato la leggenda secondo la quale i floppy disc sono tutti uguali

Difatti sembrano tutti uguali finchè non si osserva con attenzione il jacket. Qui termina l'uguaglianza.

La maggior parte delle società costruttrici sigillano i dischi un punto qui, un punto là, lasciando parte dei lembi non sigillati.

Prima o poi ai lembi accadono cose naturalissime: si gonfiano, si curvano, si raggrinziscono... in poche parole si aprono.

Con penne, matite, unghie persino un ragazzino di quattro anni come Teo può infilarsi in quegli spazi aperti.

Naturalmente è un danno enorme perchè se si inserisce qualcosa di molle e slabbrato nel disc-drive quest'ultimo può incepparsi; si può rovinare la testina e si possono perdere i dati. Questo può accadere con gli abituali sistemi di chiusura ma non con i dischetti Memorex che usa un procedimento esclusivo chiamato "Solid-Seam Bonding".

Con questo sistema ogni singolo millimetro quadrato dei lembi di tutti i dischi Memorex viene sigillato ermeticamente, rendendoli più rigidi e più resistenti.

È un sistema che consente al floppy disc di sostenere ogni assalto, che impedisce alla testina di rovinarsi e ai dati di andare perduti.

Il che sta a dimostrare che un floppy disc Memorex non è uguale a tutti gli altri: è migliore. E il sistema di saldatura è solo un esempio della cura infinita con cui viene prodotto ogni floppy disc Memorex; sia esso da 8", da 5 1/4" o il nuovo 3 1/2". Questa estrema accuratezza dà la garanzia che ogni disco Memorex è al 100% perfetto.

La prossima volta che acquistate un floppy disc - o qualche centinaio - ricordate: non tutti i dischetti sono uguali... Memorex vi mette al riparo da qualsiasi inconveniente.

è importante scegli
MEMOREX
A Burroughs Company

BURROUGHS-MEMOREX S.P.A.
Divisione Computer Media
Via Ciro Menotti, 14 Tel. 02/718551
20129 MILANO MI

Come raccogliere la spazzatura

Per far ordine e pulizia, e per elaborare correttamente le stringhe, è necessario...

di Claudio Baiocchi

Riprendiamo il discorso sulla gestione delle variabili nei computer Commodore: dopo un riepilogo sul trattamento delle variabili numeriche (articolo "La virgola è mobile", CCC N. 27), discuteremo la gestione delle variabili non numeriche, cioè delle cosiddette "stringhe".

I puntatori

Ricordiamo che il termine "puntatore" si riferisce ad una coppia di byte (celle), generalmente consecutivi, che contengono, rispettivamente, la "parte bassa" e la "parte alta" del numero "puntato" dal puntatore stesso. Così, ad esempio, quando si dice che le celle 43 e 44 "puntano" all'inizio del Basic, significa che il programma Basic ha inizio alla locazione di memoria numero:

PEEK(43) +256*PEEK(44). Sul C/128 tale puntatore si trova in 45 e 46, anzichè in 43 e 44. Anche tutti gli altri puntatori di cui parleremo sono, sul C/128, "sfasati" di due.

Su Vic 20, C/16, C/64 e Plus/4 l'area RAM a disposizione del Basic termina alla cella puntata da 55 e 56 ed è così ripartita:

- il programma Basic inizia a partire dalla cella puntata da 43, 44 e ha termine con tre zeri. Di norma l'ultimo di questi zeri è puntato da 45 e 46 ma l'utente può modificare tale puntatore nel ricorso a tecniche sofisticate (programmi L.M. incorporati, protezioni ed altro);

- le memorie 45 e 46 puntano all'inizio delle variabili, zona, questa, che termina una cella prima di quella puntata da 47 e 48 (la modifica di tale puntatore è vivamente sconsigliata);

- le memorie 47 e 48 puntano all'inizio dei cosiddetti ARRAY (detti anche VETTORI); si tratta delle variabili a indice, da dimensionare tramite DIM. Gli array terminano una cella prima di quella puntata da 49 e 50 ed anche per questo puntatore vale il consiglio di lasciarlo gestire dalle apposite routine Basic e di non modificarlo per strani esperimenti;

- tra la fine degli array, e la cella puntata da 51 e 52, c'è una zona "cuscinetto". Dopo ogni "garbage collection" (letteralmente: raccolta della spazzatura, operazione che sarà descritta tra poco) l'ampiezza di tale cuscinetto coincide col numero di byte ancora a disposizione ottenibile tramite un FRE(X). Da ricordare che sul C/128 occorre precisare FRE(1) mentre su C/64, se si ottiene un numero negativo, occorre sommare 2↑16 (PRINT FRE(0) +2↑16);

- le memorie 51 e 52 puntano all'inizio della zona di memorizzazione delle stringhe, zona il cui termine è puntato da 55, 56. Salvo qualche eccezione, che dettaglieremo tra poco, l'assegnazione di un valore ad una variabile stringa comporta l'abbassamento del puntatore 51, 52 (che viene diminuito della lunghezza della stringa stessa) e la memorizzazione dei caratteri costituenti la stringa nella parte

appena sottratta alla zona cuscinetto. La lunghezza della stringa, e l'indirizzo della cella iniziale della copia così costruita, sono memorizzati nella zona variabili o nella zona array, a seconda che l'assegnazione effettuata si riferisca ad una variabile semplice o ad un elemento di un array.

A esser sinceri, la descrizione dell'ultimo punto è molto imprecisa ma torneremo a più riprese su questa fase per fornire maggiori dettagli. Diamo, per ora, una regoletta che può accelerare l'esecuzione di un programma: quando si fa uso di array molto lunghi è bene dimensionarli DOPO aver definito tutte le variabili semplici. In altre parole, è bene attribuire un valore qualunque ad ogni variabile semplice che verrà usata nel programma, PRIMA di dimensionare gli array. Il motivo è semplice: ogni volta che viene incontrata una nuova variabile semplice occorre spostare in avanti di 7 posti tutta la zona riservata agli array... Per avere un'idea dei tempi necessari, si faccia girare il seguente programma tenendo conto che è valido per il C/128, C/64 e Plus/4. Per ciò che riguarda il Vic 20 e C/16 è opportuno diminuire il valore dell'argomento di DIMA$(7500) in modo da evitare un "Out Of Memory Error".

```
100 TI$="000000"
110 DIM A$(7500)
120 READ A,B,C,D,E,F,G,H,I,J
130 PRINT"TEMPO 1"TI
140 DATA .........
150 :
160 CLR
170 :
180 TI$="000000"
190 READ A,B,C,D,E,F,G,H,I,J
200 DIM A$(7500)
210 PRINT"TEMPO 2"TI
```

Tabella 1

Tempi rilevati
(in 60mi di secondo):

Modello	Tempo1	Tempo2
C/128 (slow)	1185	31
C/128 (fast)	569	15
Plus/4	1024	26
C/64	265	18

Il programma mostra come una non accorta gestione degli array può far perdere molto tempo: la semplice inversione delle linee 110, 120 (cfr. le 190, 200) permette di realizzare un sensibile risparmio di tempo.

Abbiamo finora descritto la situazione per quanto concerne i "piccoli" della famiglia Commodore. Sui "grandi", a partire dal C/128, la zona programma e la zona variabili si trovano su due distinti banchi di memoria, e non interferiscono tra loro. FRE(0) e FRE(1) forniscono rispettivamente lo spazio libero per la scrittura di nuove linee Basic e quello per la definizione di nuove variabili o array. In particolare, modifiche del programma NON cancellano (come invece avviene sui "piccoli"), eventuali array o variabili precedentemente definiti. Si tenga inoltre presente che sul C/128 il puntatore di inizio variabili, puntando ad un diverso banco di memoria, non fornisce la fine del programma Basic.

Scendiamo in magazzino

Per descrivere in dettaglio come il computer memorizza le stringhe, è comodo far riferimento ad un'analogia con un magazziniere, il cui problema è il seguente.

Le merci in magazzino, ognuna individuata tramite un proprio codice, necessitano di una scheda che contenga varie informazioni: numero di pezzi in magazzino, elenco fornitori, ordinativi in corso di evasione, IVA da fatturare ed altro: tutti dati, in altre parole, che necessitano di una frequente revisione. Uno dei sistemi più intuitivi per raggiungere lo scopo è quello di tipo "agenda telefonica": quando i codici sono ordinati alfabeticamente, reperire un articolo, e la corrispondente scheda, è un'operazione facile e veloce.

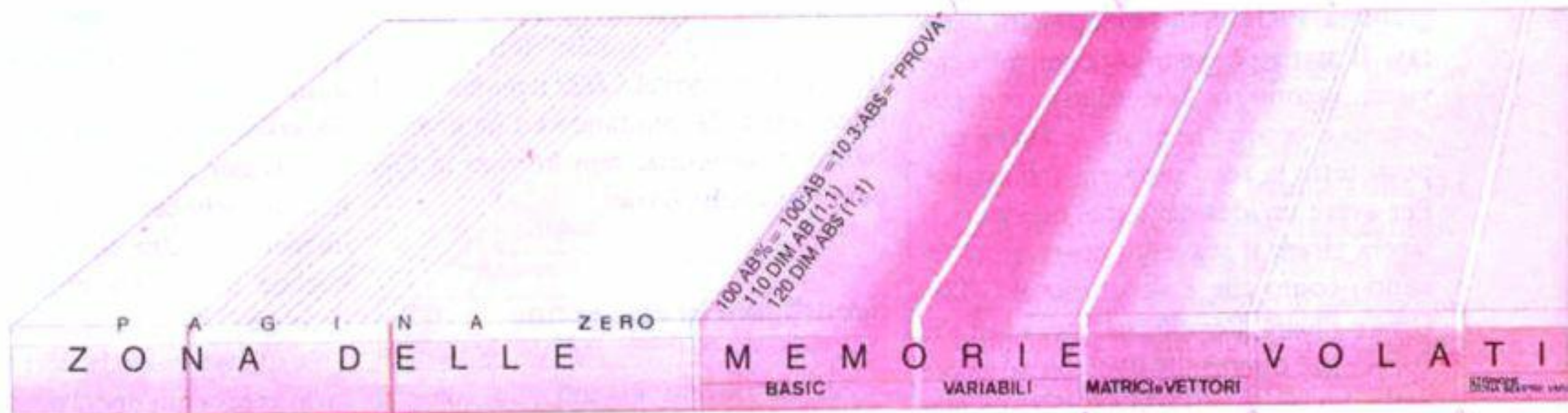
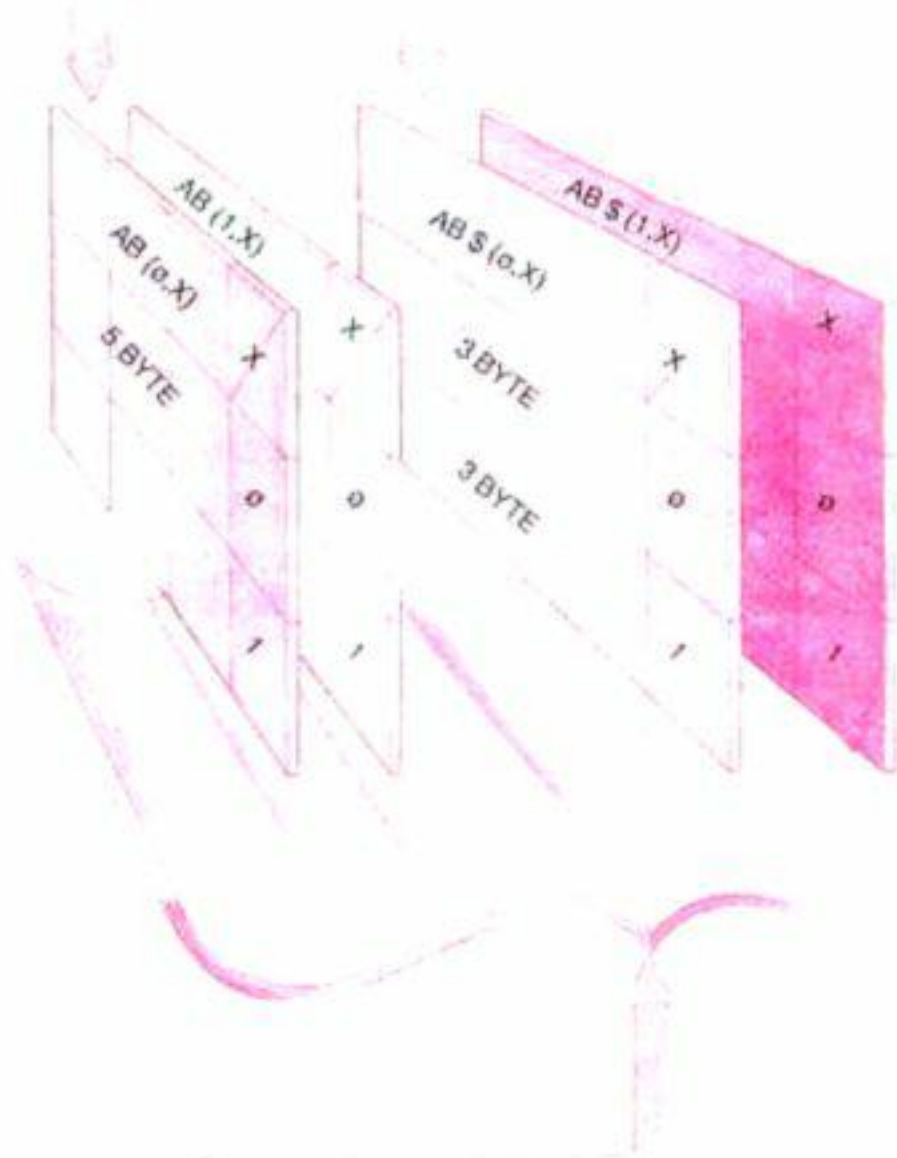
Tuttavia, con tale sistema, il nostro magazziniere si troverebbe presto nei guai; salvo il caso banale di aggiornamenti, che non modificano la lunghezza delle precedenti informazioni, ogni altra operazione provocherà problemi più o meno gravi: sostituire una scheda con una più corta, come pure sopprimere una scheda perchè l'articolo corrispondente non è più trattato, provocherà "solo" spreco di spazio (basta cancellare il testo vecchio ed eventualmente riusare una parte dello spazio per il testo nuovo).

Ma la sostituzione di un testo con uno più lungo e, peggio ancora, l'inserimento in elenco di una nuova scheda, risulteranno operazioni molto laboriose, a meno che non si voglia rinunciare all'ordine alfabetico... Ma, a pensarci bene, l'ordine alfabetico è l'unico motivo per adottare un sistema di tipo "agenda"!

Volendo dunque rinunciare all'ordine alfabetico, il sistema più naturale è quello di costruirsi un "indice". Costruiremo a parte, quindi, il vero schedario, e in un mini-schedario separato, scriveremo, a fianco ai vari codici, le pagine che la scheda corrispondente occupa realmente nello schedario. Ad esempio: pagina iniziale e pagina finale, oppure pagina iniziale e numero di pagine occupate eccetera.

In questo modo la ricerca di un codice sarà ancora un'operazione facile anche se gli articoli non sono sistemati in ordine alfabetico. Per la ricerca dovremo sfogliare solo l'indice, e non l'intero schedario. L'inserimento di nuovi articoli non provoca problemi di sorta, ma la modifica di schede già esistenti sembra riproporre gli stessi problemi di prima.

In realtà la situazione è cambiata radicalmente, nel senso che è ora perlomeno possibile "rinviare" il problema. Possiamo, cioè, procedere nel modo seguente per modificare una scheda, *lasciamo al suo posto la vecchia scheda ormai inutile*, e scriviamo la nuova scheda nella zona vuota dello schedario, modificando poi, nell'indice, le informazioni sulle pagine occupate. Si tratta però di uno spreco di spazio e, presto o tardi, dovremo sospendere ogni altra attività per riorganizzare il nostro schedario sopprimendo le pagine inutili, compattando quelle utili, ed aggiornando via via i numeri delle pagine riportati nell'indice. Il vantaggio consi-

AREA BASIC
N. DI BYTE
FUNZIONE
2
Link linea Basic successiva
2
Numerazione linea
0÷80
Istruzioni Basic
ULTIMO
Codice termine linea (0)
ULTIMI TRE
Codice termine programma (0,0,0)
BASIC
AREA VARIABILI
(7 BYTE PER CIASCUNA)
N° - BYTE
FUNZIONE
2
NOME VARIABILE
VALORE INTERO
3-4
VALORE INTERO
VALORE DECIMALE
3-7
VALORE DECIMALE
STRINGA
3+
4-5
NUMERO CARATTERI
INDIRIZZO PRIMO CARATTERE
TIPO DI VARIABILE
PRIMO BYTE
INTERA
CODICE ASCII +
DECIMALE
CODICE ASCI
STRINGA
CODICE ASCI
NOME
100 AB% = 100:AB = 10.3:AB$ = "PROVA"
110 DIM AB (1,1)
120 DIM AB$ (1,1)
P A G I N A ZERO
ZONA DELLE MEMORIE VOLATI
BASIC
VARIABILI
MATRICI e VETTORI
TAVOLA DEGLI INDIRIZZI
PEEK (X)+PEEK (X+1)*256
X
X+1
A 43 44 Inizio Basic
B 45 46 Inizio variabili
C 47 48 Inizio matrici
D 49 50 Fine matrici
E 51 52 Inizio stringhe
F 53 54 Fine stringhe
G 55 56 Fine mem. RAM

ste, però, nel fatto che fino a quando non vi saremo costretti, per mancanza di spazio, questa operazione può essere rinviata, senza con ciò compromettere l'efficienza del nostro schedario. Al contrario, nell'impostazione di tipo alfabetico, per avere uno schedario sempre efficiente siamo costretti a fare la risistemazione dopo OGNI operazione di scrittura, rallentando così enormemente il lavoro.

Naturalmente niente ci obbliga a tenere separato l'indice dallo schedario; rovesciando l'uso tradizionale, metteremo l'indice nelle PRIME pagine del nostro schedario. Se riempiamo lo schedario a partire dall'ultima pagina (e procedendo a ritroso con le schede), non ci saranno difficoltà nè ad inserire nuovi elementi, che saranno messi in coda all'indice, nè ad aggiungere le corrispondenti schede, che saranno messe in testa alla vera parte schedario. Qualche difficoltà si porrebbe a sopprimere alcune schede, ma questa operazione (comando ERASE su certi computer) non sarà presa in considerazione.

Per rendere ancora più facili gli inserimenti, aggiungeremo infine una pagina iniziale al nostro schedario, in cui annotiamo la pagina finale della parte indice e la pagina iniziale della parte schedario. In tal modo tutta la gestione sarà banale (!) finchè la coda dell'indice non si "scontrerà" con la testa dello schedario, cioè fin quando i due numeri scritti nella prima pagina non verranno a coincidere....

Ed è esattamente sul principio ora descritto che si basa la gestione delle stringhe nei computer Commodore:

- *le memorie 45 e 46 dicono dove comincia l'indice;*
- *le 49, 50 dicono dove finisce; si tratta in realtà di due indici, uno per le variabili semplici ed uno per gli array. Sia tra le variabili che tra gli array non ci sono solo le stringhe, ma il succo del discorso è identico...;*
- *le memorie 55, 56 dicono qual'è l'ultima pagina dello schedario;*
- *le 51, 52 dicono dove iniziano le vere schede; e nella zona variabili ad ogni stringa è associata la lunghezza e (se la lunghezza non è 0) il punto di inizio della "scheda" corrispondente. Sul C/128 ci si ricordi di aumentare di 2 tutti questi puntatori.*

Il disegno di queste pagine, che rende più chiara la mappa della memoria, è tratto dal N. 10 di CCC in cui fu pubblicato l'articolo "Un po' d'ordine tra i bit". In questo articolo, tra l'altro, si parlava dell'allocazione delle variabili numeriche, del loro "codice" e di esperimenti vari su variabili numeriche, stringa, vettori e matrici (un po' di pubblicità non guasta mai...)

Seguiamo ora il processo più da vicino: accendiamo il computer (o eseguiamo un CLR se avevamo già usato qualche variabile) e digitiamo:

X0$="PIPPO" (R)

L'interprete, per eseguire questo comando banale, effettua in realtà varie operazioni:

1/ controlla nella zona variabili se la stringa X0$ è già stata usata; nel nostro caso la risposta è no;

2/ memorizza nella zona variabili la variabile X0$, riservandole 7 byte (di cui solo i primi cinque sono utilizzati): nel primo deposita il valore 88, codice di X; nel secondo 176 (=48 +128. 48 è il codice di 0, 128 è caratteristico delle variabili stringa); nel terzo byte mette 5, lunghezza della stringa PIPPO ; nel quarto e quinto byte mette il puntatore alla zona in cui ricopierà la stringa PIPPO che, per ora, è presente sullo schermo e nel buffer del Basic, ma presto potrebbe scomparire da entrambi;

3/ provvede ad aumentare di 7 i puntatori di inizio e fine array; e sposta di 7 posti in avanti tutta la zona che contiene gli array (qui l'operazione è velocissima: array non ce ne sono!);

4/ abbassa di 5 (=lunghezza della stringa) il puntatore 51, 52; e ricopia nei 5 byte così "protetti" i valori 80, 73, 80, 80, 79 (codici ASCII di P,I,P,P,O)

In realtà, tanto nella fase (3) quanto nella (4) c'è un'operazione in più: il controllo che la fine degli array resti minore dell'inizio delle stringhe. Qui tutto va bene; tra poco vedremo cosa succede quando le cose vanno male. I possessori di C/16, Plus/4 e C/128 tengano inoltre presente che sui loro computer lo spazio riservato nella zona stringhe è pari alla lunghezza della stringa AUMENTATA DI 2; vedremo più avanti il perchè.

Continuiamo il nostro esperimento digitando:

PRINT PEEK(51) +256*PEEK(52)-PEEK(49)-256*PEEK(50), FRE(1) (R)

per ottenere, in due modi diversi, la quantità di memoria ancora disponibile. Naturalmente i due numeri ottenuti saranno identici (su alcuni esemplari di C/64 si sostituirà FRE(1) con: 2↑16 +FRE(1)).

Modifichiamo ora il contenuto di X0$ eseguendo:

X0$="GIUSEPPE" (R)

Questa volta nella fase (1) la risposta è positiva: l'"articolo" X0$ esiste già nel nostro schedario, e c'è solo da aggiornarne la scheda. Le fasi (2), (3) vengono saltate; effettuata la fase (4), vengono solo modificati i byte 3, 4 e 5 dei 7 che descrivono X0$.

Controlliamo ancora la quantità di memoria disponibile eseguendo:

PRINT PEEK(51) +256*PEEK(52)-PEEK(49)-256*PEEK(50), FRE(1) (R)

otterremo stavolta due numeri diversi...

Il perchè è ovvio: se chiediamo al computer la quantità di memoria ancora a disposizione, la prima operazione eseguita è la "risistemazione" dello schedario, onde fornirci la "vera" risposta: qui si tratta solo di sopprimere la ormai inutile stringa PIPPO, spostare verso l'alto la stringa GIUSEPPE e modificare il puntatore di X0$; solo al termine di questa operazione (che i francofoni chiamano "ramasser les ordures", e gli anglofoni "garbage collection") verrà effettuato il conteggio della quantità di memoria ancora disponibile, tramite appunto la formula PEEK(51) + 256*PEEK(52) - PEEK(49) - 256*PEEK(50).

In realtà, quando il numero di variabili su cui lavorare aumenta (almeno su Vic 20 e C/64) raccogliere la spazzatura non è

una cosa semplice, perchè i caratteri che compongono le stringhe non hanno un contrassegno che distingua le parti "attive" da quelle ormai inutili: su C/16, Plus/4 e C/128 le cose vanno molto meglio: come già detto, su questi modelli la memorizzazione di una stringa richiede due byte in più della lunghezza della stringa stessa: in questi due spazi viene memorizzato un puntatore alla stringa che usa tali caratteri. Quando la stringa viene modificata i due spazi in più vengono usati per segnalare l'inutilità dell'informazione. Grazie a questo accorgimento i tempi di esecuzione dell'operazione di pulizia risultano assolutamente trascurabili, sia pure al prezzo di un leggero rallentamento della velocità necessaria per effettuare le assegnazioni.

Dalla teoria alla pratica

Per controllare le affermazioni qualitative fin qui fatte, organizziamo un "match" tra i vari modelli. Il primo round sarà uno "scontro spaziale": la sfida è descritta qui di seguito ed è nettamente a favore della "vecchia gestione". Si faccia attenzione al trattamento speciale riservato da Vic 20 e C/64 alle stringhe inserite nel testo: benchè permetta di risparmiare quantità notevoli di spazio, tale gestione può risultare pericolosa in occasione di un CHAINING, cioè di un programma che, ad esecuzione terminata, ne chiama un altro; ad evitare spiacevoli incidenti, se i due programmi in questione fossero:

```
10 REM PRIMA PARTE
20 A$="GIUSEPPE"
30 READ B$
40 DATA PIPPO
50 LOAD "SECONDA PARTE"
```

```
10 REM SECONDA PARTE
20 PRINT A$
30 PRINT B$
```

sarà bene sostituire le linee 20 e 30 della prima parte ad esempio con:

```
20 A$="GIU" +"SEPPE"
30 READ B$:B$ =
   LEFT$(B$,1) +MID$(B$,2)
```

Per maggiori chiarimenti su quest'ultima parte, rinviamo il lettore all'articolo "Tecniche di Overlay" pubblicato su CCC N.9 che tratta in modo approfondito e completo il problema del caricamento di un programma durante l'esecuzione di un secondo programma Basic (e per oggi basta con la pubblicità...).

Il diverso metodo di gestione delle stringhe adottato sui modelli Commodore più recenti (C/16, Plus/4, C/128) fa si che programmi perfettamente funzionanti su Vic 20 inespanso (cioè con soli 3.5 K di RAM utente) generino un "Out Of Memory" su C/16 (16 K di RAM utente), su Plus/4 (64 K di RAM utente) e su C/128 (128 K di RAM utente). Un esempio banale di programma che "sta" in un Vic 20 inespanso ma genera un "Out Of Memory" anche sul C/128 è il seguente:

```
10 X=0:DIM A$(1001)
15 FOR X=0 TO 1000 STEP 2
20 READ A$(X):RESTORE
25 A$(X+1)="CHI TROPPO IN ALTO
   SAL CADE SOVENTE PRECIPITE
   VOLISSIMEVOLMENTE"
30 NEXT:DATA 33 TRENTINI ENTRA
   RONO TUTTI E 33 TROTTERELLA
   NDO IN TRENTO
```

In effetti su Vic 20 e C/64 le stringhe definite nel programma, o inserite in esso come DATA e lette tramite READ, non occupano spazio supplementare nella "zona stringhe". Al contrario, su C/16, Plus/4 e C/128, ogni stringa, anche se inserita nel programma Basic o letta come DATA, viene ricopiata nella zona stringhe. Le stringhe di lunghezza L>0 occupano in tale zona uno spazio di L +2 byte.

Velocità di esecuzione

Il secondo round dello scontro sarà organizzato sui tempi di esecuzione necessari ad effettuare le assegnazioni. Qui di seguito descriviamo lo scontro che si risolve ancora in favore della vecchia gestione, anche se i tempi necessari per il calcolo della memoria libera risultano preoccupanti...

Si tenga presente che, grosso modo, su Vic 20 e C/64 i tempi impiegati per l'operazione di pulizia dipendono essenzialmente dal numero di stringhe presenti nell'indice (ma quelle nulle e quelle "inserite nel testo" contano molto meno...) e, purtroppo, da un lato questa dipendenza è QUADRATICA (con buona approssimazione: sul C/64 il tempo richiesto per fare pulizia su N variabili stringa non nulle è di circa N*N/12000−N/720 secondi), e d'altro lato il fatto di avere "poca spazzatura" non accelera affatto l'operazione. Al contrario, con la nuova gestione, il tempo per la pulizia cresce LINEARMENTE ed è comunque molto basso; anche se, come si è già visto, il guadagno nei tempi si paga salato in termini di spazio occupato.

Digitate il seguente programma che si limita a dimensionare il vettore A$, assegnare (in modo un po' tortuoso) la stringa "**" ai suoi 4001 elementi, e cronometrare il tempo impiegato :

```
10 TI$="000000":X=0:DIM A$(400
   0)
20 FOR X=1 TO 4000:A$(X)="*"+"
   *":NEXT
30 PRINTTI$
```

Date il RUN, e controllate che, a seconda del computer a disposizione, i tempi si mantengono entro limiti più che accettabili: 22 secondi su C/64, 28 secondi su Plus/4 e 35 secondi su C/128 (sul C/16 compare l'ovvio Out Of Memory).

Digitate ora:

TI$="000000": X=FRE(1): PRINTTI$ seguito dal solito (R). La risposta arriva in soli tre secondi su Plus/4 e in 4 secondi sul C/128. Se, invece, state lavorando sul C/64, rinunciate a pestare disperatamente (e inutilmente) il tasto Run/Stop: potete tranquillamente scendere al Bar e prendervi un buon caffè! Eppure il computer non è guasto: ha semplicemente bisogno di OLTRE 22 MINUTI per fare "pulizia"!

Si tenga presente che sul C/64 il tempo necessario per l'operazione di pulizia non dipende dalla quantità di memoria libera nè dalla quantità di "sporcizia" da

eliminare: se dopo l'esperimento precedente si esegue una seconda volta il comando FRE (ed ormai la spazzatura è stata raccolta tutta...) si dovranno attendere altri 22 minuti prima di rivedere il cursore lampeggiante sotto la scritta READY!

I risultati del secondo round potrebbero far pensare che la vecchia gestione sia comunque superiore: basta disinteressarsi della quantità di memoria libera e non effettuare il comando FRE... In realtà le cose non stanno così, ed il terzo round lo dimostra: quando la zona apparentemente libera risulta insufficiente per nuove assegnazioni, il computer è costretto ad eseguire l'operazione di pulizia, anche in assenza del comando FRE; e questa volta la lentezza dell'esecuzione si ripercuote sul tempo totale necessario per l'assegnazione!

Strano ma vero

Quando un programma crea "spazzatura" può presentarsi per il computer la necessità di fare pulizia durante l'elaborazione del programma stesso. Ciò non è un problema su C/16, Plus/4 e C/128 (dove la raccolta della spazzatura è molto veloce) ma rischia di provocare attese lunghissime sul C/64 e sul Vic 20 espanso. Il seguente programma, ad esempio, chiede un numero N per dimensionare il vettore A$(N); poi fa scattare il cronometro ed assegna ad ogni elemento del vettore una stringa composta da 5 caratteri casuali:

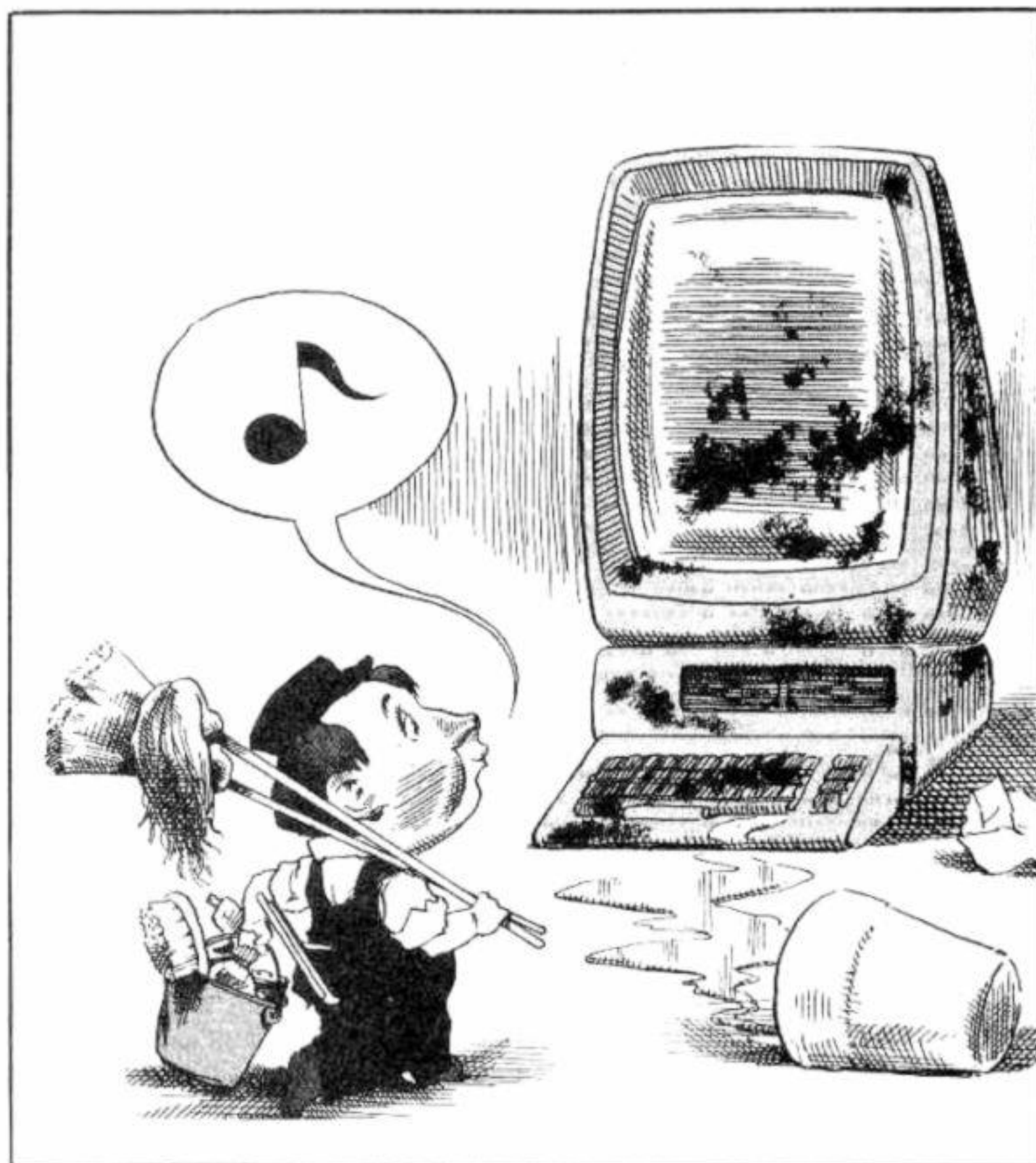
```
100 A$="":Y=0:X=0:A$=""
110 READ NMAX:IF FRE(0)<0 THEN
    READ NMAX
120 PRINT"QUANTI ELEMENTI":INPU
    T N
130 IF N<1 OR N>NMAX THEN END
140 DIM A$(N):TI$="000000"
150 FOR X=1 TO N:A$=CHR$(256*RN
    D(1))
160 FOR Y=1 TO 4:A$=CHR$(256*RN
    D(1))+A$
170 NEXT:A$(X)=A$:NEXT
180 Y=TI/60:PRINT"TEMPO"TI$
190 PRINT"VELOCITA':"INT(.5+N/Y
    )"ELEMENTI AL SECONDO": RUN
200 DATA 6400:REM NMAX SU 128
210 DATA 4750:REM NMAX PER C/64
```

Oltre alle quantità indipendenti da N, il programma necessita di 3*N byte nella zona variabili; nella zona stringhe i byte necessari sono 5*N su C/64 e 7*N su C/128. Occorrerà pertanto tenere N inferiore a circa 4800 su C/64, ed inferiore a circa 6420 su C/128; la linea 110 provvede, a seconda del computer su cui sta girando il programma, a fissare un valore NMAX che eviterà un Out Of Memory.

In realtà i problemi cominciano con N ben lontano da NMAX: per ogni X la costruzione di A$(X) genera un spazzatura (19 byte su C/64, 37 su C/128) col risultato che, per N ancora lontani da NMAX, il computer dovrà sospendere la sua attività e dedicarsi ad un'operazione di pulizia. La seguente tabella (incompleta: non ho avuto il coraggio di testare su C/64 valori di N vicini ad NMAX!) mostra che ciò non è un problema sul C/128, mentre ha conseguenze letali sul C/64. Nella tabella le velocità VEL sono espresse in numero di elementi riempiti al secondo:

Tabella 2

N	(tra 20 e 1400)	(1500)	(2500)	NMAX
Vel C64	12	5	3	???
Vel C128	10	10	10	9

Arrivederci a presto per trattare altri argomenti causa di grattacapi...

OPERAZIONE FEDELTÀ SYSTEMS.

I VINCITORI!

Pubblichiamo l'elenco dei quaranta lettori che sono risultati vincenti alla seconda estrazione mensile del concorso "Vince chi legge". L'estrazione è stata effettuata il 28 aprile 1986 presso la sede della Systems, alla presenza di un funzionario dell'Amministrazione finanziaria.

I primi cinque estratti vincono una stampante Commodore a margherita; dal sesto al decimo estratto riceveranno una printer-plotter Commodore; dall'undicesimo al quarantesimo un libro a scelta della biblioteca Systems oppure un abbonamento a una rivista Systems. Tutti i coupons ricevuti concorreranno all'estrazione del premio finale, consistente in una moto Cagiva Electra 125cc.

Ed ecco i primi quaranta nomi (tra parentesi il numero d'ordine del coupon di partecipazione):

1 Marinello Lorenzo - Eraclea (VE) - (272)
2 Piamonti Maurizio - Lucca -(482)
3 Lillini Adriana - Roma -(68)
4 Pelle Alessio - Genova -(873)
5 Mantovani Maurizio - Ferrara - (737)

6 De Luca Luigi - Napoli - (304)
7 Spandaro Luca - Udine - (662)
8 De Felici Raffaele - Frascati (Roma) - (86)
9 Gherardi Daniele - Bosco M. (Ferrara) - (270)
10 Volpe Massimo - Milano - (214)

11 Milano Andrea - Agrigento - (424)
12 Di Nocera Umberto - C/Mare Di Stabia (NA) -(895)
13 Santostasi Sergio - Bari - (250)
14 Chechi Andrea - Sarteano (SI) - (19)
15 Bussola Renato - S. Pietro in Cariano (VR) - (843)
16 Ronchiato Marico - Jesolo (VE) - (629)
17 Tomasella Miguel Angel - S. Vendemiano (TV) - (388)
18 Lieto Luciano - Ciampino (Roma) - (677)
19 Cerulli Mario - Milano - (293)
20 Caffiero Antonio - Cagliari - (122)
21 Baglione Antonio - Palermo - (943)
22 Gava Tiziano - Treviso - (762)
23 Olgiati Fabio - Dairago (MI) - (162)
24 Zecchini Erio - Milano - (215)
25 Dragone Cosimo - Falconara M. (AN) - (638)
26 Esposito Gualtiero - Salerno - (314)
27 Sottilotta Franco - Marina di Ravenna - (350)
28 Mariani Giovanni - Bologna - (591)
29 Genero Marcello - Savona (929)
30 Quinti Marco - Milano - (764)
31 Fineschi Massimo - Firenze - (444)
32 Daglio Claudio - Collegno (TO) - (610)
33 Maragon Eriberto - Loreo (RO) - (418)
34 De Bernardi Aldo - Sampierdarena (GE) - (380)
35 Palermo Vincenzo - Napoli - (612)
36 Lissona Luca - Origgio (VA) - (679)
37 Proivitera Aurelio - Zafferana Etnea (CT) - (385)
38 Salzano Renato - Napoli - (52)
39 Paolini Massimo - Milano - (246)
40 Guerrini Roberto - Roma - (563)

Speciale Sport

a cura di Marco Miotti

Da qualche tempo presentiamo su C.C.C. uno "speciale" dedicato ai migliori videogiochi che hanno per tema un particolare argomento.

Dopo "Giochi di guerra" (apparso sul numero 30), presentiamo ora uno "speciale sport" dedicato ai più diffusi giochi sportivi, e precisamente:

1 - Match point: sicuramente uno fra i migliori Tennis per C/64.

2 - Ping pong: il popolare tennis da tavolo giocato in maniera eccezionale dal computer.

3 - International soccer: il classico calcio.

4 - Exploding fist: Arti marziali dai colpi mortali.

5 - One on one: l'eccezionale Basket per C/64.

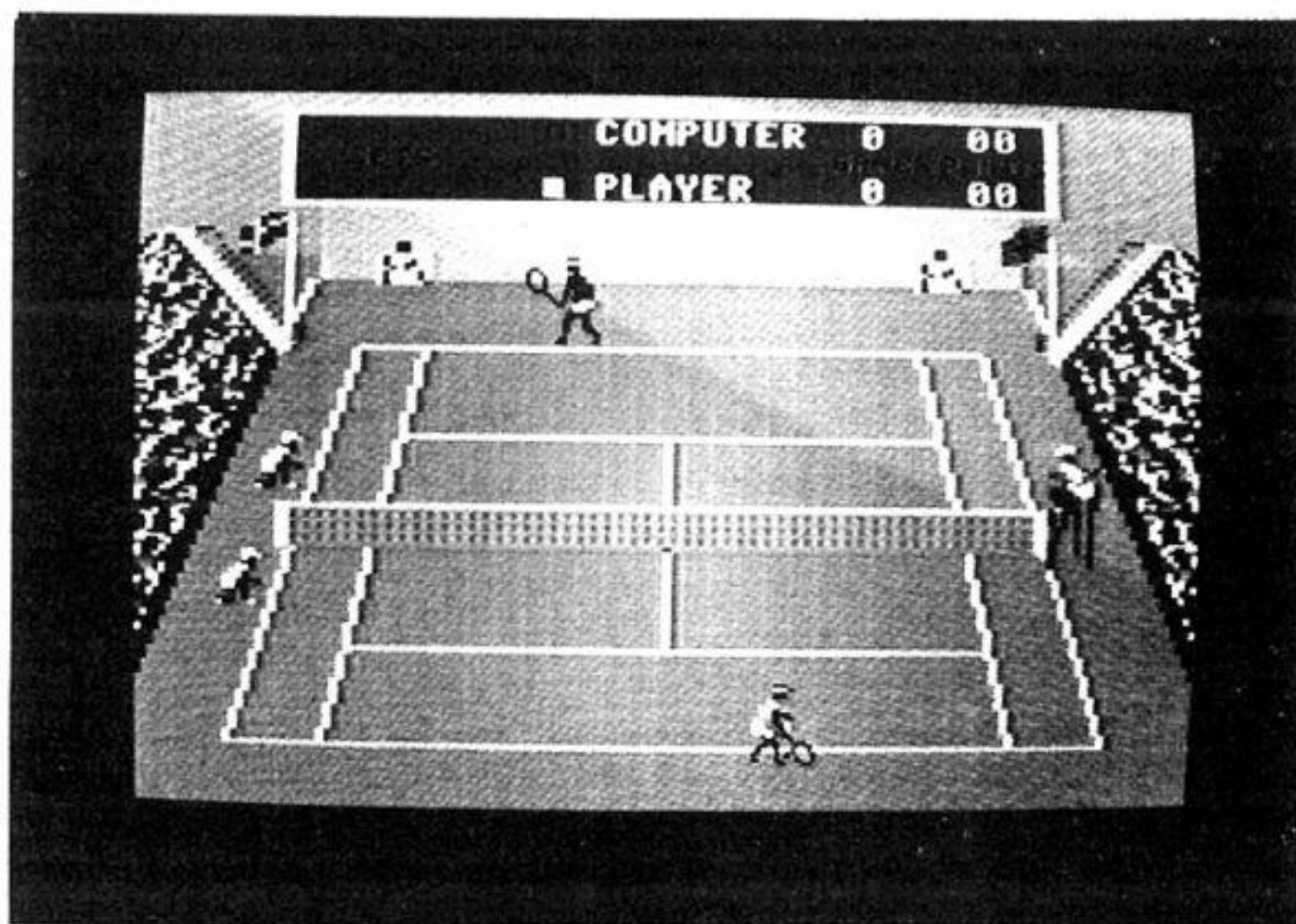
*** * ***

Queste recenzioni sono dedicate a tutti coloro che pur possedendo uno o più giochi fra quelli descritti non sono in grado di capirne appieno il funzionamento.

Match Point

Sarete in grado di battere a tennis il vostro C/64? Forse non subito, ma dopo un breve allenamento forse sì.

Che cosa significa match point?

Dipende: per i giocatori di tennis il Match point è il momento culminante in cui si decide la vittoria di uno dei due giocatori. Per il Commodoriano appassionato di videogiochi, invece, Match point è un tennis elettronico che rappresenta l'apice dell'evoluzione tennistica, nel campo dei computer, iniziata quasi dieci anno fa con il famosissimo "tennis", gioco in cui due racchette stilizzate dovevano respingere una pallina da una o dall'altra parte del campo.

Da allora è stata fatta molta strada ma le vere novità nel campo hanno stentato a mostrarsi.

Match point rappresenta quindi un punto di arrivo, una sosta obbligata per colui che fa incetta di videogiochi.

A nessuno sarà sfuggita la ricchezza dei particolari quale il tabellone, il raccattapalle

Pique
commodore
Wilson
K2
SPALDING

che corre attraverso il campo restando chinato per non disturbare (o forse per non prendersi qualche palla in un occhio) e altri particolari che conferiscono un realismo eccezionale al gioco.

Il programma, non appena lanciato, visualizza un menu iniziale in cui scegliere se giocare ai quarti di finale, alla semifinale o alla finale; occorre anche decidere se agire contro il computer oppure se giocare in due persone servendosi di due joystick.

Fatte le scelte opportune, ci si trova immediatamente nel campo da gioco, attorniati da una folla attenta e silenziosa in attesa dell'inizio; i colpi della palla mettono in risalto il religioso, quasi mistico silenzio (bella questa frase vero? Ragazzi, questa è poesia!). Occorre ora lanciare in alto la palla, colpendola con tutta la forza possibile, per dirigerla nel campo avversario.

La partita avrà termine quando uno vincerà (bella scoperta): come accade nel vero tennis non esiste infatti limite di tempo e in casi estremi una partita può durare ore ed ore fino ad un massimo di sei set (e sono tanti per un computer miserello, quasi francescano come il C 64).

Le regole sono le stesse del tennis "vero" tanto che ci sembra superfluo ricordarle; non sperate, comunque, di imbrogliare l'arbitro dato che, come tutti i cervelli elettronici, è particolarmente preciso e pignolo: non si lascerà sfuggire neanche un out.

I movimenti da conferire al nostro omino sono semplici e complicati allo stesso tempo: semplici, in quanto le otto direzioni ne controllano il movimento e il tasto Fire, invece, sferra fantozziane racchettate utili a colpire la palla. Potrà sembrar strano che il giocatore, dopo un po', si giri su se stesso, ma capirete presto che questo serve per dare un tocco realistico al gioco. Se, infatti, è necessario compiere piccoli spostamenti, si può anche indietreggiare di qualche passo; ma se, come nel caso del nostro tennista, dovete spostarvi per tutto il campo, sarebbe stupido (o perlomeno scomodo) viaggiare a ritroso... anche i gamberi giocano a tennis? Giuro che non lo sapevo.

Questi semplici controlli possono talvolta essere complicati da controllare e richiedono, in ogni caso, notevole coordinamento e prontezza di riflessi. Capita spesso che a poco più di un millimetro dalla palla il giocatore si giri repentinamente facendovi perdere punto e staffe; allora imparerete che è preferibile, ad un unico spostamento lungo tutto il campo, una serie di piccoli movimenti capaci di generare rapidi aggiornamenti sulla posizione, sullo stato dell'omino e sulla condizione del gioco lasciando invariato l'assetto, se così si può chiamare, dell'omino stesso.

Imparerete anche che premendo il tasto Fire e muovendo la leva in una direzione otterrete un effetto non individuabile a priori, in quanto dipende largamente dalla posizione della racchetta, dalla velocità di movimento e di rotazione della pallina, dal suo verso di rotazione e dalla posizione e velocità del giocatore: particolari che possono sembrare insignificanti ma che racchiudono il segreto del gioco.

Il vero nocciolo del problema è la dominazione dei sistemi che lo circondano, che in questo caso sono rappresentati dall'insieme delle probabili traiettorie percorribili da una pallina. Malgrado ciò possiamo comunque dare un paio di indicazioni utili per sfruttare al meglio le opportunità che si presentano nel corso della partita.

- *Premendo il tasto Fire e muovendo il joystick in avanti avremo una palla con una parabola molto larga e bassa, ciò che si definisce una schiacciata; è molto facile mandare la palla in rete utilizzando questo tipo di colpo, per cui fate attenzione ad usarlo.*
- *Schiacciando il Fire e tirando il joystick si può realizzare un bel pallonetto che viene utilizzato generalmente per spiazzare l'avversario che si trova sottorete per un attacco; generalmente è un colpo molto efficace, ma terribilmente lento, a causa dell'altissima parabola descritta dalla palla.*
- *Portando il joystick a destra o a sinistra, durante la pressione del tasto Fire, si ottengono numerosi effetti, a seconda dei parametri specificati in precedenza.*

Lo stesso discorso vale quando si è in battuta: la pressione del tasto Fire farà sì che il giocatore lanci in alto la palla che verrà colpita dalla racchetta seguendo gli effetti dettati dalle condizioni sopracitate.

Quando un giocatore si trova nell'altra metà campo si vedrà ovviamente invertire i comandi, visto che si osserva il campo da un unico punto.

La velocità del gioco non cambia sostanzialmente all'interno di uno stesso livello di difficoltà, ma si fa sentire quando questo aumenta e si passa, ad esempio, dalla lentezza esasperante dei quarti di finale alla velocità della semifinale, per finire con la ipervelocità della finale.

La tattica migliore per mettere a segno il maggior numero di punti è quella di spiazzare l'avversario se giocate in difesa e di anticiparlo se in attacco.

Per farlo, portatevi sotto rete immediatamente dopo aver toccato la palla, sia che sia stata una battuta che una ricezione; potrete così prender facilmente le palle rilanciate e mandarle dove più vi aggrada. Due sono i vantaggi nel giocare in questa zona del campo. Il primo dipende dal fatto che le palle che pervengono all'avversario possono essere sia forti che angolate, il che metterebbe una seria ipoteca sulle possibilità di successo del vostro antagonista; la seconda dipende dal fatto che qualora vi sfuggisse una palla potrete sempre tentare di riprenderla, sempre che si tratti di una palla lenta.

Esistono però anche gli svantaggi: se ad esempio vi trovate sulla destra del campo ed il vostro avversario sferra un colpo angolato completamente a sinistra... potete rinunciare ad accaparrarvi quel punto dato che prima che l'omino raggiunga la palla avete tutto il tempo di bere un caffè. Se, inoltre, l'avversario realizza un pallonetto, non avete possibilità migliori, per cui tutto sommato cercate di rimanere in difesa per quanto possibile. In tale posizione dovete tendere a spiazzare l'avversario facendolo correre da una parte o dall'altra del campo, finchè non sarà più in grado di raccogliere la sferzata decisiva; qui nè le palle alte nè quelle lunghe vi faranno paura e potrete prendere di tutto, a meno che non siate... spiazzati, come talvolta capita se non si è lesti.

Anche in questo caso i consigli da dare sono gli stessi di tutti gli altri game: un gioco è bello se diverte; cercate quindi di applicarvi al massimo, non tentate azioni di forza e attaccate solo quando siete sul sicuro. Una serie di colpi diretti allo stesso punto fanno cadere il calcolatore in una illusoria sicurezza per cui un colpo deciso improvvisamente nella direzione opposta dovrebbe (quasi) assicurarvi il punteggio.

Su battuta è difficile incrementare il proprio punteggio: quel vecchio marpione di computer prende proprio tutto; comunque qualche volta si riesce a ingannarlo.

Cercate di non giocare mai in attacco quando siete in tie break dato che siete più vulnerabili in un momento così delicato e poco conveniente; per evitare infine le fastidiose giravolte del simpatico omino è sufficiente fargli dare un colpo a vuoto prima di iniziare il "viaggio"; forse tale azione farà perdere del tempo, ma almeno sarete sicuri di non incappare in un momento girellone del nostro eroe.

E se per caso arriverete a vincere la coppa Davis ditecelo... che ci proveremo anche noi.

Ping pong

Un gioco molto veloce che darà parecchio filo da torcere anche al concorrente dai riflessi più pronti.

In Giappone una ne fanno, cento ne pensano e mille (programmi) producono.

La ventata di innovazioni introdotte dalle case nipponiche negli ultimi anni non si ferma alle stupende animazioni dei cartoni animati ed alla miniaturizzazione di tutto lo scibile umano (alberi compresi): lo scienziato giapponese guarda avanti più di molti altri ed è appunto da questo protendersi verso il futuro che nascono le principali idee non strettamente riguardanti le tecniche di programmazione.

E di tecnica di programmazione devono averne avuta parecchia per condensare quel meraviglioso gioco che è il ping pong nei freddi circuiti dell'home computer più venduto nel mondo.

Avrete già capito che si tratta del ping pong della Konami, apparso nelle sale giochi da qualche mese ed ora disponibile sul mercato software internazionale. Questo prodotto aveva già suscitato l'attenzione e l'interesse degli specialisti in... arte elettronica, considerandolo unico nel suo genere grazie alle straordinarie caratteristiche di movimento e realismo.

L'avvento del tennis da tavolo su Commodore 64 non può che destare profonda ammirazione e grande interesse per quello sport giocato più o meno da tutti almeno una volta nella vita, ma visto sempre con grande sufficienza. Molti lo considerano un semplice passatempo ignorando l'esistenza di un campionato mondiale quasi sempre vinto dagli orientali.

Quali siano le caratteristiche di un gioco di successo è molto difficile dirlo: principalmente deve piacere, essere semplice da capire e consentire un gran numero di sviluppi e di variazioni dipendenti largamente dal giocatore.

Il ping pong sembra possedere le tre caratteristiche grazie alla varietà dei movimenti della pallina che assicura un'inesauribile varietà di azioni; i movimenti si riducono a non più di quattro o cinque, ma tutti da scoprire.

Rammentiamo brevemente le principali regole del ping pong:

- *Il campo è rettangolare e diviso in quattro parti uguali. Le due metà campo sono divise da una rete alta circa quindici centimetri e sporgente alle estremità.*
- *Lo scopo del gioco è quello di toccare la palla lanciata dall'avversario (dopo che avrà toccato una sola volta il nostro campo) e rilanciarla dall'altra parte senza toccare il nostro campo. Chi non intercetta la palla, oppure la lascia uscire, fa guadagnare un punto all'avversario.*

Come spiegazione potrà sembrare sommaria, ma purtroppo non può essere affrontata in dettaglio in questa sede.

Il videogioco, graficamente, si presenta bello e spiritoso: una gradevole musichetta introduttiva mostra un tavolo da ping pong sormontato dal messaggio "Ping Pong" ed un grazioso aimaletto a forma di pera intento a guardare il giocatore.

Ecco che arriva la pallina che, passata dietro le lettere componenti il messaggio, trapassa la lettera "O" e cade sul tavolo, rimbalzando su tutto ciò che sembra solido per poi cadere sulla testa del grazioso peroide (o pinguino?) che, dopo un attimo di smarrimento cade svenuto mentre la palla esce di scena rotolando.

Le prime volte che si gioca, è utile osservare il modo in cui si comporta il computer. Se, infatti, non si preme alcun tasto comincia una vera e propria partita dimostrativa in cui il calcolatore gioca continuamente contro se stesso fino alla realizzazione del punteggio. Premendo il tasto Fire del joystick (in porta due) si dà inizio al gioco.

Un minimo menu di scelta appare ora sullo schermo: muovendo il joystick è possibile scegliere il livello di difficoltà variabile da uno a quattro (seguite il nostro consiglio disinteressato: non siate precipitosi e cominciate dal più basso).

Al primo livello le cose non saranno affatto difficili e il computer fornirà palline a velocità moderata. Un avversario piuttosto imbranato vi porrà su un piatto d'argento un mucchio di pallonetti approfittando dei quali potrete improvvisare "schiacciate" vertiginose. Non è necessario, inoltre, spostarsi lingo il bordo del tavolo perchè il movimento è totalmente automatico; dovete limitarvi a schiacciare il tasto Fire solo quando arriva la

pallina.

Il secondo livello è un tantino più difficile: la pallina si muove più velocemente ed il computer ha finalmente pensato di ingaggiare un giocatore più bravo. Anche qui dovrete manovrare abilmente per tirare in trappola il computer che, ovviamente, cercherà di offrire un maggior numero di "risposte" valide.

E', invece, un vero giocatore professionista l'avversario del terzo livello: cercherà anche lui, infatti, di "schiacciare" e sarà più difficile sopraffarlo. Come se non bastasse le (poche) schiacciate che riuscirete a sferrare saranno quasi sempre abilmente parate e respinte dall'agguerritissimo avversario che respingerà una palla fortemente smorzata, costringendovi ad alzare il tiro: da attaccante diventerete vittima del computer.

Al quarto livello di difficoltà vi sembrerà di avere di fronte il campione universale di ping pong.

Le palline arrivano ad una velocità apocalittica (e al terzo livello osavamo chiamarle veloci!): il computer para quasi tutti i tiri e ne mette a segno un'infinità...

Quando ci è nata la malsana idea di affrontare il computer al massimo livello di difficoltà (il quinto) abbiamo dovuto subito desistere a causa della reale impossibilità perfino ad iniziare il gioco.

Se, quindi, ci riuscite voi, saremo ben contenti di ricevere i vostri suggerimenti per avere una minima probabilità di successo: noi non ci siamo riusciti.

Se siete appassionati di grafica sostate un minutino di fronte alla ricchezza di particolari di cui è dotato il gioco, tanto tocca a voi battere il primo colpo ed il programma non ha alcun controllo sul tempo massimo di attesa.

Noterete immediatamente la buona resa prospettica, quasi come se osservaste la scena utilizzando una macchina fotografica dotata di obiettivo grandangolare; il pubblico è lì che aspetta (anche se, a voler esser pignoli, sembrano più un branco di scimmie che un gruppo di spettatori).

I movimenti della racchetta sono i seguenti:

- *Spingendo in avanti la leva del joystick si genera un movimento rotatorio che conferisce alla pallina l'energia sufficiente per una schiacciata.*
- *Tirando il joystick si può osservare che la racchetta assume una posizione a scudo, che serve per "smorzare" tiri pericolsi.*
- *Premendo il tasto Fire si ottiene il movimento più comune nel ping pong: il palleggio.*
- *Tirando la leva a destra o a sinistra si orienta la racchetta nella corrispondente direzione.*
- *Il movimento del giocatore è automatico e seguirà sempre la direzione della palla consentendovi di concentrarvi sui suoi movimenti.*
- *La battuta viene effettuata in seguito ad una combinazione di movimenti da parte del giocatore: spingendo prima la leva, e tirandola in seguito verso di sè, può rappresentare spesso una soluzione valida in risposta a tiri difficili. Spingendo la leva si può osservare che la racchetta si muove dal suo punto di riposo e si sposta verso l'alto; il secondo movimento deve essere effettuato prima che la racchetta ritorni al punto di partenza altrimenti occorrerà ripetere il tiro. Questo, probabilmente, è stato fatto per consentire un'ampia possibilità di decisione sul come, dove e quando effettuare la battuta.*

Superata la rete, dopo la battuta, l'avversario rimanderà indietro la palla, che dovrete intercettare e respingere. Il punteggio verrà assegnato al giocatore che tirerà la pallina in modo da impedire all'avversario di prenderla.

Le palla viene battuta cinque volte per ciascuno; non ha importanza chi realizza, o meno, il punto. In generale chi si trova alla battuta è avvantaggiato rispetto all'altro, ma non si può mai dire l'ultima parola.

Occorre molta pratica per riuscire in giochi di destrezza come il Ping Pong, ma un paio di consigli possiamo sempre darli:

- *Una alzata involontaria da parte di uno qualsiasi dei contendenti è sempre segnalata da un profondo muggito; questo è il momento migliore per schiacciare.*
- *Per indurre l'avversario ad alzare la palla occorre inviarne una molto lenta; così facendo l'avversario sarà costretto ad alzare la palla per non farla andare in rete. Inutile dire che questo serve per aprirvi la strada a successive schiacciate.*
- *Un lungo palleggio, pur se spettacolare, è altrettanto pericoloso in quanto un effetto impresso alla palla potrebbe costringervi ad una pericolosa alzata con rischiose conseguenze.*
- *Per conferire effetti speciali alla pallina, giratevi da una parte o dall'altra a seconda che la palla provenga da destra o da sinistra e cercate di spiazzare l'avversario.*
- *Una buona combinazione di smorzate ed effetti indurrà quasi sempre l'avversario ad alzare la palla; approffittatene per schiacciare. Ricordatevi che ad una schiacciata si può rispondere solo con una smorzata e che smorzare una schiacciata (scusate il bisticcio di parole) significa alzare notevolmente la palla; questo fatto vi porrà ancora nelle condizioni favorevoli per una nuova schiacciata.*
- *Giocate molto sullo spiazzamento dell'avversario al terzo e quarto livello; i risultati si faranno vedere.*

Un ultimo, doveroso consiglio: in caso di insuccesso non sbattete perterra il joystick: ricordatevi che è solo un gioco!

International soccer

Durante i Mondiali in Messico non poteva mancare un articolo sul più diffuso videogame ispirato al calcio.

Quattro anni fa undici uomini vestiti di azzurro conquistavano il titolo di campioni del mondo in quello sport giustamente riconosciuto come il più popolare ed appassionante in assoluto: il calcio.

Passato il periodo di sosta (se così si può chiamare), l'Italia scende nuovamente in campo contro le rappresentanze degli altri Paesi del mondo; l'Italia è ora, più che mai, il Paese da battere.

Messico '86. Se ne parla dappertutto; chi polemicamente, chi appassionatamente chi... elettronicamente.

Il calcio elettronico per il Commodore 64 ha certamente fatto storia; mai un gioco per home computer torna a far parlar di sè dopo quasi tre anni dalla sua uscita, in seguito al precedente mondiale di Spagna 1982. Che cos'è, dunque, che rende un programma del genere entusiasmante e richiesto ancora a distanza di anni dalla sua commercializzazione?

I maligni diranno che dipende dal fatto che nessuno si è mai preoccupato di crearne una versione migliore; ma coloro che hanno seguito dal principio la storia del Commodore 64 sanno con certezza quando un programma vale veramente e quando un altro è solo una velocissima meteora.

Capita molto spesso, infatti, di rispolverare vecchi giochi come International Soccer magari dimenticati in cassette polverose.

Interrompiamo qui la rievocazione della epopea del C'64, superata in lunghezza solo dagli antichi testi sumerici, e passiamo alla descrizione delle regole del gioco, sicuramente dimenticate dai più.

Non appena caricato, e posto in esecuzione, il programma presenta la scelta delle squadre che avviene cambiando il colore delle maglie ai giocatori visualizzati ai due lati dello schermo (non aspettatevi di trovare tutte le squadre presenti al Mundial; i colori disponibili sul Commodore 64 sono soltanto sedici); chi possiede due joysticks, oltre ad un

compagno-avversario, può tranquillamente accedere al gioco premendo il tasto Fire del joystick in porta 2. Chi, invece, desiderasse misurarsi contro il computer, deve impostare il livello di difficoltà altrimenti vedrà in campo una squadra di "statue", dato che il computer sarà alla ricerca di un altro giocatore fantasma e l'unico ad avere in mano l'iniziativa sarà il giocatore in carne ed ossa.

I livelli di difficoltà, numerati da uno a nove, si differenziano tra loro nella possibilità di andare a segno: questa è inversamente proporzionale al livello scelto e se è la prima volta che giocate tenete quest'ultimo piuttosto basso, almeno all'inizio.

I giocatori del computer, in genere, sono molto più coordinati dei vostri da cui discende la regola numero uno del buon bit-dipendente: non scoraggiarsi mai. Imparerete ben presto a saltare gli schemi tattici del calcolatore che risulta, fortunatamente, privo di quel fattore umano che caratterizza lo svolgimento di qualsiasi competizione.

Ultima caratteristica del menu di scelta è la possibilità di scegliere tra un televisore (o monitor) a colori oppure uno in bianco e nero: la scelta si imposta premendo il tasto F7 che seleziona, se premuto nuovamente, una delle due possibilità.

Operate quindi nel seguente modo:

- *Scegliete il colore delle maglie tenendo conto che la vostra squadra, se giocate contro il computer, è quella che si trova alla destra dei teleschermi; queste scelte vengono fatte tramite i tasti F1 per la squadra di destra e F3 per quella di sinistra.*
- *Premete il tasto F5 se intendete giocare contro il computer; il giocatore di sinistra sparirà per lasciare il posto al giocatore numero uno che rappresenta il livello selezionato; per modificarlo premete più volte il tasto F5 fino ad ottenere quello desiderato.*
- *Se avete il televisore a colori premete il Fire del joystick in porta due, altrimenti provate a premere F7; se il risultato vi aggrada potete incominciare a giocare, altrimenti premete più volte il tasto F7; se, comunque, non riuscite a distinguere bene i vostri uomini dagli avversari, limitatevi a controllare il gioco basandovi sui colori delle maglie.*

E' giunto il momento magico: le due squa-

dre escono ordinatamente dagli spogliatoi e si avviano alle rispettive metà del campo.

Ma... un momento: i giocatori dovrebbero esser undici; come mai qui se ne vedono solo nove?

Questo è, fortunatamente, l'unico aspetto che limita il realismo del gioco in questione; tuttavia, pensando a quanto grande sia un vero campo di calcio e quanto poco può essere visualizzato su un monitor...

Che cosa, insomma, si può pretendere da un home computer? Rammentiamo, inoltre, che il numero di sprite visualizzabili contemporaneamente è soltanto otto a meno di non ricorrere a sofisticate tecniche di gestione delle interruzioni.

Superato lo shock da carenza di uomini, possiamo subito renderci conto del realismo del gioco corredato dai fischi dell'arbitro (che non si vede ma c'è), dagli applausi e da un realistico rumore prodotto dal rimbalzo della palla sul terreno.

I movimenti del giocatore sono controllati dal joystick (ma guarda un po') da spostare nelle otto direzioni possibili; la pressione del tasto Fire determina un violento calcio alla palla che, allontanatasi dall'area di controllo del giocatore, si porta qualche metro più avanti. Risulta così possibile eseguire passaggi e tiri in otto direzioni diverse.

Il tasto del joystick serve anche per "scartare" l'avversario: se premuto al punto giusto gli si porta via la palla che viene quindi in possesso del giocatore controllato dal joystick. Attenzione però che per sottrarre la palla occorre essere in contatto diretto con l'avversario e che il computer, da buon calciatore, non si farà portar via il pallone senza reagire cercando di riprenderlo a tutti i costi.

Ultima funzione del tasto Fire è quella di consentire al giocatore che rimette in gioco la palla (in seguito ad un fallo laterale o durante una rimessa da fondocampo) di lanciarla in direzione di un compagno controllato dal joystick.

Come è possibile controllare tutti i giocatori presenti in campo mediante un solo joystick?

L'ostacolo è stato aggirato in modo ingegnoso: tutti i giocatori vengono controllati dal computer tranne due che indossano maglie più scure. Questi seguiranno le direzioni imposte loro dai joystick, nel caso giochino due persone, oppure da un joy e dal computer.

Anche la selezione dei due giocatori è effettuata in modo semplice ed efficace:

- *Se la palla è stata toccata da un giocatore inquadrato nello schermo, egli sarà individuato dal colore della maglia, che risulta la più scura in campo.*
- *Nel caso, invece, in cui nessuno dei giocatori (di una stessa squadra) abbia toccato la palla oppure, dopo averla toccata, "esce" dallo schermo, il giocatore "capo" sarà il più vicino alla sfera.*

Tutti gli altri giocatori non vengono assolutamente controllati anche se tendono a dirigersi verso la palla e ad allontanarsene non appena un compagno ne guadagna il possesso (per disporsi meglio, tatticamente).

Ciò avviene sempre ad eccezione di tre circostanze:

- *La palla entra in fallo laterale: il joystick controlla il giocatore (con la maglia più scura) che la riceve mentre il tasto Fire consente al compagno incaricato della rimessa di tirare il pallone.*
- *La palla è sul fondo e agli attaccanti spetta il calcio d'angolo: vengono seguite le stesse modalità rispettate nel caso precedente; la palla, in seguito ad una lunga e bassa parabola, arriverà in prossimità dell'attaccante più scuro che potrà, eventualmente, segnare.*
- *Durante un'azione in difesa la sfera viene lanciata dagli avversari in direzione della nostra porta. Per parare il tiro occorre muovere il portiere in vari modi: premendo semplicemente il Fire si otterrà un salto verso l'alto; premendo il Fire e spingendo il joystick in alto si avrà un tuffo alla sua destra o alla sua sinistra, a seconda che questo sia nella parte destra o sinistra della porta; premendo il tasto e tirando la leva si ottiene l'operazione opposta.*

Ad eccezione dell'ultimo caso, in cui il portiere resterà fermo, il computer provvederà a rimettere in gioco la palla se il giocatore non opererà per tempo; evidentemente non sono previsti i minuti di recupero per il tempo perso.

La partita è composta da due tempi di duecento secondi l'uno al termine dei quali vincerà la squadra che avrà realizzato il maggior numero di reti.

All'inizio il computer si trova alla destra del teleschermo, mentre voi sarete a sinistra; finito il primo tempo le due squadre invertiranno le proprie posizioni.

La migliore tattica da seguire durante le fasi del gioco è quella di giocare "ragionando": una lunga serie di passaggi è da preferirsi rispetto ad una lenta e pericolosa discesa che ci renderà facili prede dei centrocampisti avversari; i tiri angolati sono più difficili da prendere ed il portiere avversario, se comandato dal computer, gioca a livelli... elettronici, parando palloni inafferrabili da un semplice umano. E' tuttavia possibile raggirarlo facendogli credere di essere in procinto di tirare: il computer tende infatti a lanciarsi anticipando il giocatore. Facendo tesoro di quanto affermato non è eccessivamente difficile segnare un goal.

Un tiro da fuori area arriva difficilmente in porta ma, quando arriva, segna quasi sempre. Ciò è dovuto al fatto che il portiere cerca di anticipare il giocatore, per cui, tirando da lontano, si riesce a rendere vano il suo tentativo. Potrebbe però esservi un avversario nelle vicinanze che, sentito odore di pericolo, non esiterà ad impossessarsi della palla.

Per sottrarre il pallone ad un avversario, occorre mettersi sulla stessa traiettoria, raggiungerlo e premere il pulsante per impossessarsi della sfera. Occorre però a questo punto cambiare fulmineamente la direzione onde evitare che il computer metta in pratica la stessa tattica.

I colpi di testa in prossimità della porta sono generalmente molto efficaci e si ottengono premendo il tasto Fire durante un lancio lungo nel caso in cui la palla entri in contatto col crapone del giocatore come, ad esempio, durante un corner.

Un ultimo consiglio che diamo è quello di non seguire ostinatamente la palla, se posseduta dall'avversario, oppure nel caso in cui il giocatore che controllate si trova molto distante dal pallone. In questo caso ricordate che i vostri compagni tendono sempre (o quasi) ad avvicinarsi alla palla e che uno di loro assumerà il vostro posto se voi non sarete presenti nelle vicinanze.

Provate quindi a dirigervi all'esterno dello schermo (ovviamente a destra o a sinistra) e vedrete cosa succede.

Questi ultimi non sono consigli ma trucchi che abbiamo scoperto giocando: nei livelli da uno a quattro il computer non effettua quasi mai passaggi lunghi ed inoltre spesso lancia la palla in fallo laterale o direttamente sul fondo; questo non avviene nei livelli da cinque a nove in cui, al contrario, il computer sembra giocare in modo piuttosto razionale. In questi casi giocate... irrazionalmente cercando di disorientarlo. Questa tecnica, a lungo andare, fornisce i suoi frutti. Provate, tra l'altro, a roteare vorticosamente il joystick quando sarete nell' area avversaria e vedrete che sarà più semplice aver ragione dell'avversario.

Cos'altro si potrebbe dire su questo gioco che non sia ancora stato detto?

Una cosa c'è, ed è il finale; ma questo è meglio che lo scopriate da voi, cercando di vincere la partita, se vi riesce!

Jiulius Erwing and Larry Bird go one on one

Per chi è appassionato di Basket uno stupendo gioco ispirato ai campioni.

"In direta from Pasadeina Chelifournia Iu Es Ei trasmettiamo la finale En Bi Ei tra Boston Seltics and Filadelfia Seventi Sicsers..."

Quante volte abbiamo sentito (pronunciate meglio) frasi come questa dai più famosi commentatori televisivi su una delle partite più spettacolari di Basket?

Bene, la possibilità di rivivere le entusiasmanti sfide tra i campioni NBA viene offerta dalla Electronic Arts, già famosa per avere prodotto programmi del calibro di "Seven Cities Of Gold" e "Heart Of Africa". Ripropone ora, a distanza di due anni dalla prima uscita, il famoso gioco della pallacanestro elettronica "One on one" che accese a suo tempo notevoli entusiasmi sia per l'eccezionale grafica e suoni, sia per la quasi totale fedeltà di riproduzione dei movimenti dei due campioni: Larry Bird dei Celtics e Julius Erwing, in arte Doctor "J" dei 76ers.

Possiamo quindi ammirare i due campioni in fantastici duelli e numeri spettacolari che ci terranno col fiato sospeso.

Le regole sono note a tutti e ci limiteremo a fare soltanto alcune precisazioni:

- *Il tiro a canestro nel campionato NBA deve essere compiuto entro 24 secondi dall'inizio dell'azione, pena la perdita della palla (in Italia il tempo è di 30 secondi).*
- *La linea dei tre punti si trova a oltre sette metri dal canestro.*
- *Avete a disposizione tre time out per ogni partita; usati saggiamente costituiscono un'ottima arma contro il computer.*
- *I falli che possono essere commessi sono gli stessi della pallacanestro reale (sfondamento, passi, eccetera); dopo cinque falli subiti un giocatore avrà un bonus; ad ogni fallo farà seguito l'intervento dell'arbitro che motiverà le sue decisioni ed assegnerà i tiri liberi al giocatore che ha subito il fallo.*
- *Se nella foga tirerete giù il canestro non preoccupatevi: vi beccherete solo una sgridata dall'operaio di turno che si vedrà costretto a pulire per terra.*

Una volta caricato il gioco, ponetelo in esecuzione e davanti ai vostri occhi vedrete apparire i due rivali Dr. "J" e L. Bird impegnati in un autentico corpo a corpo dimostrativo ed accompagnati dalla famosa "Maple Leaf Rag"; a questo punto potete accedere al menu di scelta con cui programmerete la giornata sportiva premendo il tasto F1.

Apparirà una schermata con numerosi messaggi che indicano le opzioni attivabili.

Il tasto F3 disattiva la musica e gli effetti sonori.

Premendo F7 vedremo il cursore lampeggiante muoversi verso la riga successiva.

Premendo F5 vedremo che il lampeggio si sposta nel resto della riga.

Il tasto F7 viene stavolta utilizzato per muoversi nei campi all'interno di una riga; premendo nuovamente F5 si ritornerà ad agire sul cursore lampeggiante di sinistra.

Tutto questo può essere sostituito dal joystick in porta uno: il cursore verticale viene controllato dai movimenti del joystick in alto e in basso; premendo il pulsante di sparo si controlla la riga a destra del cursore lampeggiante e ci si potrà muovere spostando il joystick a destra o a sinistra.

Una volta operate le scelte premete nuovamente il tasto Fire per posizionarvi nuovamente sul cursore verticale.

Descriviamo ora le principali funzioni delle opzioni selezionabili.

Per quanto riguarda la prima riga:

- *Resume game: serve per riprendere il gioco nel punto in cui era stato interrotto (un gioco viene interrotto mediante la pressione del tasto F1).*
- *New game: scegliendo questa opzione inizia un nuovo gioco, seguendo tutte le direttive specificate in precedenza.*
- *Demo: questa opzione permette di osservare come giocano i due contendenti.*

Relativamente alla seconda riga abbiamo:

- *Park and rec: questo è il livello di difficoltà più basso.*
- *Varsity: in questo livello, più difficile del primo, l'arbitro comincia a fischiare i falli e le cose si fanno più complesse.*
- *College: la maggior parte dei campioni sono nati sportivamente nei college universitari; qui si è ad un passo dall'essere professionisti per cui avrete una vita grama.*

La terza linea fornisce informazioni sui due giocatori.

- *2 player: si gioca in due (con due joystick).*
- *Computer Dr. J: il computer assume le sembianze di Erwing e giocherà contro Bird impersonato da voi.*
- *Computer Larry Bird: stavolta sarete voi a giocare con Dr. J.*

La quarta riga offre le seguenti possibilità:

- *Timed game: qui si accede ad un sottomenu col quale si sceglie la durata del quarto del gioco: 2, 4, 6 oppure 8 minuti.*
- *Play to set score: esiste anche la possibilità di giocare fino al raggiungimento di un punteggio da selezionare mediante un altro sottomenu.*

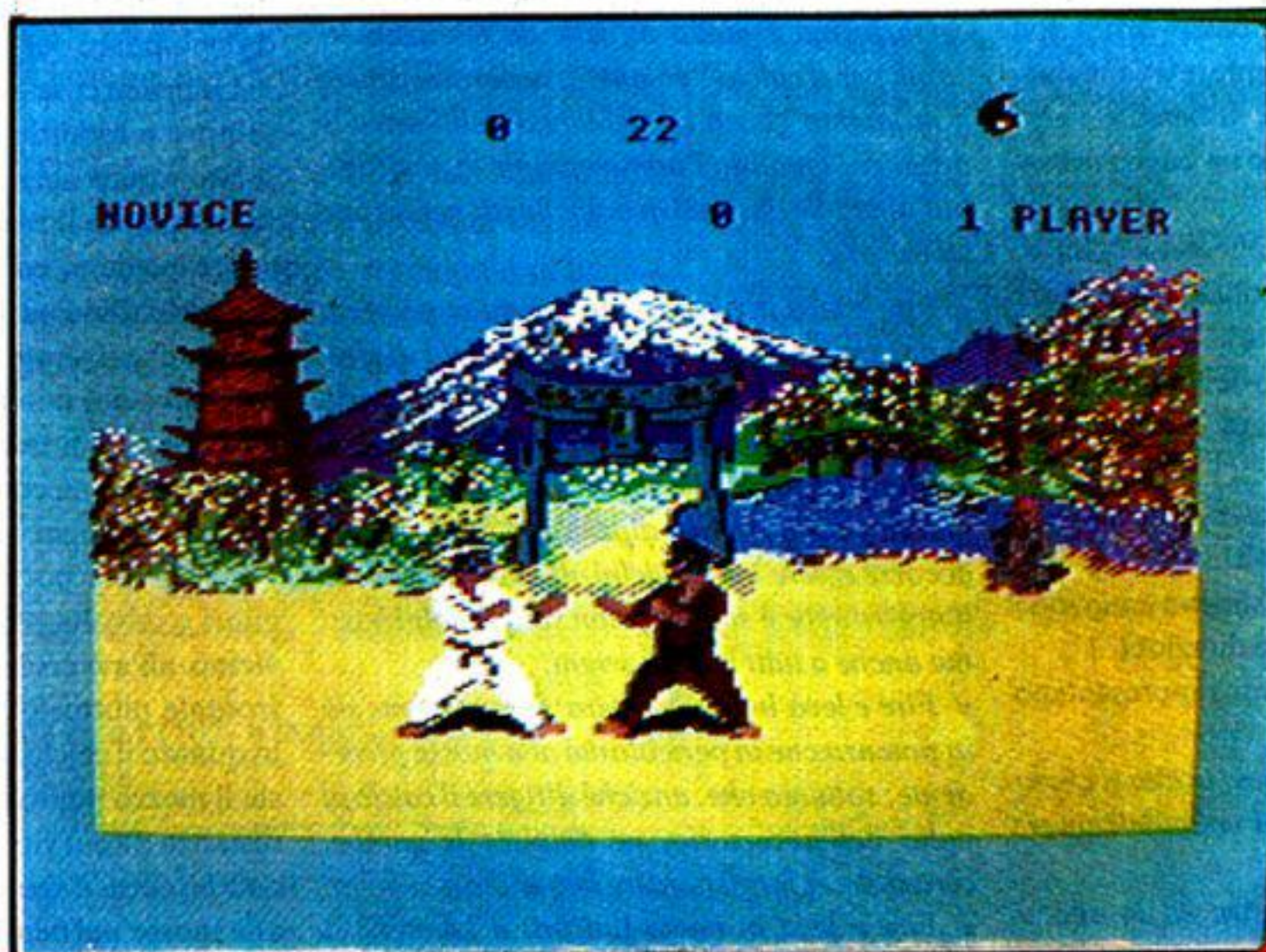
L'ultima riga stabilisce a chi assegnare la palla in seguito al conseguimento di uno o due punti.

Il controllo dell'omino è relativamente semplice: sono accettate le otto direzioni del joystick, a meno che sulla strada non si trovi l'avversario.

Per quanto riguarda il tasto Fire del joystick, diremo che premendolo per un breve istante faremo voltare l'uomo mentre, tenendolo premuto a lungo, effettuerà un tiro o salterà a canestro se si trova sotto rete;

Le differenze tra i giocatori sono sostanzialmente due e riguardano diverse caratteristiche quali la difesa e precisione (tipiche di Bird) e la potenza in attacco (propria di Dr. J).

The way of the exploding fist

Un videogame per chi è dotato di nervi d'acciaio, pelle dura e prontezza di riflessi.

Un appassionato di arti marziali non può restare indifferente di fronte ad un simile programma.

"The way of the exploding fist" è sicuramente il miglior prodotto del genere che, affiancato da prodotti validi ma di qualità decisamente inferiore (come Karateka, Bruce Lee ed altri), rappresenta l'apice del gioco d'azione e di tattica, condito, com'è, da ottima grafica e musica di sottofondo.

Chi conosce la perigliosa via di un esperto in lotte orientali saprà certamente quanto tempo, pazienza e applicazione occorrano per giungere all'ambita cintura nera. Non temete, però, dato che nel gioco non occorre seguire l'intera carriera, bensì cercare di raggranellare il maggior numero di punti possibile avendo di già, come "base", la cintura nera.

Chi volesse misurarsi con un amico può farlo grazie alla possibilità di selezionare l'avversario: computer oppure secondo giocatore.

Come il computer abbia potuto acquisire una tale tecnica di gioco è un vero mistero: passando di livello in livello ci si trova davanti avversari sempre più esperti ed astuti che riescono quasi sempre ad anticipare le nostre mosse.

Dovremo affrontare i nemici in quattro diversi scenari oltre a quello di "bonus" in cui bisognerà abbattere un possente toro. In ciascuno di questi scenari dovremo battere il campione per ben due volte prima di passare al successivo. In caso di sconfitta, il gioco finirà.

I colpi vengono conteggiati in base ad un criterio di pericolosità ed efficacia. Si possono avere diverse opportunità che assegnano un punto intero, o metà, a seconda dei colpi inferti (o subiti). Per vincere la manche occorre realizzare due punti; nel caso la manche finisse senza l'abbattimento di uno dei due concorrenti (occorre infatti terminare in un tempo limite), la vittoria andrà a colui che possiede più punti; in caso di parità la vittoria verrà assegnata al difensore, cioè il computer.

Quando si gioca in due ogni colpo mandato a segno determina l'abbattimento di uno dei due contendenti, mentre il passaggio alla fase successiva avviene solo al termine del tempo limite.

L'eventuale vittoria dei concorrenti verrà decretata da un baffuto arbitro che alzerà una bandierina del colore corrispondente a quello del vincitore: bianco per il giocatore e rosso per il computer.

Partito il programma, si potranno ammirare le mosse e gli scenari disponibili grazie ad un confronto dimostrativo. In seguito potremo scegliere, premendo il tasto F3, le modalità dell'incontro, vale a dire se giocare ad uno o a due giocatori. Una volta effettuate le scelte, col tasto F1 inizia il gioco.

Osservando il demo si può immediatamente notare la complessità delle mosse effettuabili tanto da pensare che sarebbe impossibile giocare con un solo joystick. E' invece possibile selezionare ben 17 mosse, nove delle quali orientando il joystick in una delle otto direzioni, e le altre otto unendo al movimento della leva la pressione del pulsante di sparo.

Nel primo caso le mosse sono nove (e non otto) in quanto la direzione basso-destra non è utilizzabile da sola, ma è sempre da anticipare con un movimento verso il basso o verso destra, mosse che provocano un pugno in basso oppure in ginocchio.

I movimenti fondamentali sono quattro: uno per ogni direzione principale del joystick (senza premere il tasto Fire). Spingendo la leva in alto si otterrà un salto dell'omino particolarmente utile per evitare i colpi bassi dell'avversario.

Tirando la leva verso il basso il giocatore si abbasserà per evitare colpi alti.

I movimenti a destra e sinistra muovono l'omino nelle corrispondenti direzioni.

Gli altri cinque movimenti rappresentano mosse di attacco:

- *Spingendo la leva a destra ed in alto si sferra un possente pugno diretto al volto dell'avversario.*
- *Spingendo la leva a sinistra ed in alto si ottiene il salto con capriola verso destra.*
- *Portando la leva nella posizione basso sinistra otterremo una capriola con salto verso sinistra.*
- *La combinazione destra destra-basso sferra un pugno diretto al ventre del nemico.*
- *La combinazione basso destra-basso fa abbassare l'uomo (per parare un colpo alto) e gli fa sferrare un pugno sempre al ventre, restando però stavolta inginocchiato.*

Premendo il tasto Fire del joystick si dipone di altre otto mosse:

- *Fire e leva in avanti: l'uomo spicca un agile balzo in avanti e sferra uno dei più potenti calci consentiti dal gioco, ma il calcolo della giusta distanza rende insicuro l'utilizzo di questa mossa che risulta essere comunque di notevole efficacia.*
- *Fire e leva in basso: l'uomo, appoggiandosi alle mani, si flette e sferra un calcio ai piedi dell'avversario provocando un dispettoso sgambetto; questa mossa è di gran lunga più sicura della precedente, ma molto più debole.*
- *Fire e leva a destra (oppure da solo): calcio in avanti che raggiunge il massimo della potenza quasi alla fine della corsa: ne consegue che più vicino sarà l'avversario, meno efficace risulterà il calcio e tanto meno varrà per il conteggio dei punti. Oltre che per attaccare, questo calcio serve per mantenere a debita distanza l'avversario con l'unico svantaggio di rendere il nostro eroe sensibile e fragile ai colpi bassi.*
- *Fire e leva a sinistra. E' certamente la mossa più spettacolare del gioco: l'omino compie una fantastica giravolta su se stesso e sferra un calcio al capo dell'antagonista. Normalmente questa mossa fa guadagnare un punto all'avversario ma è molto difficile da realizzare e, come per il calcio "volante", occorre scegliere il momento giusto e calcolare al millimetro le distanze. Quando l'uomo compie la piroetta è più sensibile che mai ai colpi bassi; perciò fate attenzione.*
- *Fire e leva in alto-destra: questa mossa è tra le più sicure del gioco; consente infatti di sferrare un calcio direttamente al volto dell'avversario restando, come si suol dire, con i piedi per terra. I rischi che si corrono nell'utilizzare questo tipo di mossa dipendono dal fatto che occorre essere molto vicini all'avversario e che il nostro eroe è soggetto non solo ai colpi bassi ma anche a tutti i tipi di pugni.*
- *Fire e leva in basso-destra: è equivalente sia in potenza che in pericolosità alla mossa precedente; soltanto che, anzichè dirigere il calcio al volto dell'avversario, ne colpisce le gambe, cercando di fargli perdere l'equilibrio.*
- *Fire e leva in basso-sinistra: il karateka si abbassa e sferra un calcio a terra ed all'indietro mossa, questa, che si rivela particolarmente utile quando ci si trova l'avversario alle spalle. Da usare soprattutto in seguito ad un salto, è anche molto efficace in altre occasioni dato che i due uomini sono indifesi alle spalle e possono schivare i colpi soltanto eseguendo salti o abbassandosi.*
- *Fire e leva in alto-sinistra: da usare solo quando il giocatore si trova alle spalle del nemico. Questo colpo sferra un poderoso e preciso calcio diretto al suo volto. La pericolosità di questa mossa, e della precedente, consiste nel fatto che può essere utilizzata per prevenire la precedente, e viceversa.*

Esaminate le mosse che i due contendenti possono compiere, aggiungiamo che quanto detto vale per il giocatore che si trova con il volto rivolto verso la destra dello schermo; per quanto riguarda l'altro, nel caso non fosse anche lui rivolto nella medesima direzione, occorre invertire tutte le mosse escluse quelle che hanno le direzioni alto e basso. Con un po' di pratica vi accorgerete che la gestione del gioco è più semplice di quanto non si creda; non preoccupatevi quindi per i primi inevitabili insuccessi.

Per voltare l'omino dalla parte opposta occorre muovere la leva del joystick nella direzione desiderata e tenere premuto a lungo il tasto Fire, fino a quando, cioè, il protagonista non si sarà posizionato come desideriamo.

Indicare strategie vincenti per "Exploding" è piuttosto difficile perchè il computer non sempre si comporta in modo standard di fronte a situazioni identiche, per cui è indispensabile limitarsi ad illustrare alcune idee generali da sviluppare per proprio conto.

La prima cosa da tenere in mente è che non sempre è redditizio ostinarsi a totalizzare un punto intero anzichè mezzo in una sola mossa, in quanto il computer gioca molto bene in contropiede e, come nella boxe, un pugno o un calcio d'incontro fanno... molto male e poi finisce che il punto lo fa il computer. Un colpo ben assestato non fa guadagnare molto, mentre occorre puntare a non perdere tempo prezioso che viene conteggiato in ragione di cento punti per ogni secondo risparmiato.

I colpi più potenti non sempre sono i migliori e talvolta portano il nostro eroe troppo vicino all'avversario; è preferibile un calcio frontale piuttosto che un calcio con piroetta, in quanto il calcio frontale può far guadagnare sia il mezzo punto sia il punto, a seconda della distanza da cui viene sferrato; mentre il calcio con piroetta è molto lento e lascia più spazio alle mosse del nemico.

Un'ottima tattica può essere quella di portarsi, mediante un balzo a capriola, alle spalle dell'avversario e sferrargli un calcio all'indietro; tale azione, tra l'altro, accresce nettamente la spettacolarità della partita.

La combinazione destra destra basso, quella che sferra un pugno in ginocchio, non fallisce quasi mai se utilizzata alla giusta distanza; usandola, però, ripetutamente, il computer prenderà gli opportuni provvedimenti sferrandovi un calcione basso.

Il modo migliore per agire è quinfi di utilizzare varie mosse di apertura e ogni tanto la mossa risolutrice che consisterà o in un calcio frontale o in un pugno inginocchiato, a seconda dei casi.

Vinca comunque il migliore e... ici ankhu misulù vatane sciabila sa!

Cosa vuol dire non lo sappiamo, ma forse vi porterà fortuna e riuscirete ad arrivare al toro.

Enciclopedia di routine

a cura di Alessandro de Simone

11100 (A proposito...)

Nell'eseguire la routine 11100 delle funzioni inverse (Sec, Csc, Cot) riportata sul N. 26 di C.C.C. ho riscontrato un errore nel calcolo della cotangente di 90 gradi (il cui valore esatto dovrebbe esser zero).

Con la routine 11100, infatti, appare sullo schermo il messaggio "ERR" con a fianco un valore approssimativo di 1.5, oppure, se il calcolo citato è preceduto dal calcolo della cotangente di un qualsiasi valore angolare, si ha come risposta, per la Cot(90 gradi), la stessa risultante dall'operazione che l'ha preceduta.

Ho appurato che ciò dipende dalla posizione matematica effettuata nella riga 11130. Tale errore si può eliminare cambiando la riga stessa come segue:

11130 IF X1$="COT" AND Y0<> ∅ THEN
X1=Y1/Y0:X5=1

Di conseguenza, nella riga 11105, il termine:
IF Y1 THEN Y2=TAN(X1)
risulta superfluo.

La correzione che suggerisco è resa possibile dall'espressione:
COT(X1)=1/TAN(X1)=COS(X1)/SIN(X1)
che, come è noto, è l'espressione goniometrica della cotangente. Naturalmente, effettuata la sostituzione di cui sopra, la routine 11100 "gira" senza più incorrere in alcun errore.

Rossano Perotti

13200 Centratrice messaggi

(Stampante MPS/803)

La routine pubblicata qui di seguito, da utilizzare con la stampante MPS/803, consente di "centrare" una qualsiasi frase su un foglio di carta.

Esistono quattro possibilità di output, a seconda del valore assegnato alla variabile X:

X=1: stampa in modo normale
X=2: stampa in "reverse"
X=3: stampa in "allargato"
X=4: stampa in allargato e in reverse

La variabile X$ contiene la stringa da stampare: questa non deve superare gli 80 caratteri, se ad "X" si assegna il valore 1 oppure 2, mentre non deve superare i 40 caratteri nel modo X=3 e X=4.

Giancarlo Castagna

```
    1 REM  ESEMPIO D'USO
    2 REM  CENTRATITRICE MPS 803
    4 :
  100 OPEN 4,4
  105 FOR I=1 TO 4
  110 X$=" COMMODORE COMPUTER CLU
      B ":X=I
  120 GOSUB 13200
  130 PRINT#4:PRINT#4:NEXT
  140 CLOSE 4
  200 :
 9999 END
13200 X0=LEN(X$):X1=INT((80-X0)/2
      )*6:X3=INT(X1/256):X2=X1-X3
      *256
13202 X1$=CHR$(14):X2$=CHR$(15):X
      3$=CHR$(16):X4$=CHR$(18)
13205 X5$=CHR$(27):X6$=CHR$(146)
13210 ON XGOTO 13220,13230,13240,
      13260
13220 PRINT#4,X5$;X3$;CHR$(X3)CHR
      $(X2);X$:RETURN
13230 PRINT#4,X4$;X5$;X3$;CHR$(X3
      )CHR$(X2);X$;X6$:RETURN
13240 X0=LEN(X$)*2:X1=(INT(80-X0)
      /2)*6:X3=INT(X1/256):X2=X1-
      X3*256
13250 PRINT#4,X1$;X5$;X3$;CHR$(X3
      )CHR$(X2);X$;X2$:RETURN
```

```
13260 X0=LEN(X$)*2:X1=(INT(80-X0)
      /2)*6:X3=INT(X1/256):X2=X1-
      X3*256
13270 PRINT#4,X4$;X1$;X5$;X3$;CHR
      $(X3)CHR$(X2);X$;X2$;X6$:RE
      TURN
13290 REM  X$= STRINGA DA CENTRAR
      E ; X= VARIABILE STAMPA
13295 REM  X=1 NORMALE ; X=2 REVE
      RSE
13298 REM  X=3 ALLUNGATO X=4 ALL.
       REV.
13299 REM  CENTRATURA MESSAGGI
```

14000 Gestione nome disco

(Per qualsiasi Commodore e drive 1541)

Come sicuramente avete intuito dal titolo, questa routine serve per leggere (o cambiare) il nome di un floppy disk, operazione poco agevole.

La lettura di un disco porta alla conseguenza di cancellare il programma in memoria mentre, d'altra parte, la formattazione implica la perdita di tutte le informazioni presenti sul dischetto.

Come è noto, il nome del disco risiede nella traccia 18 settore 0 e si estende per 16 caratteri a partire dal 144-esimo byte. Nel caso di un nome con meno di 16 caratteri, lo spazio viene riempito con il necessario numero di spazi shiftati (ASCII 160). La routine non fa altro che leggere, oppure riscrivere, i 16 caratteri utilizzando i comandi di accesso diretto del 1541.

L'uso della routine è molto semplice: si assegna un qualsiasi valore diverso da zero alla variabile X4 e si accede subito dopo alla subroutine. Al suo ritorno la variabile X1$ conterrà il nome del disco inserito nel drive.

Se, invece, X4 vale zero è necessario assegnare alla stringa X1$ il nuovo nome e "chiamare" il sottoprogramma (mediante GOSUB). Al suo ritorno X2 ed X0$ conterranno rispettivamente il codice ed il testo dell'errore che eventualmente si è verificato (esempio: dischetto con tacca di protezione coperta).

Si potrebbe, utilizzando lo stesso procedimento, modificare anche l'ID del floppy, ma... non si può. L'identificatore, infatti, viene scritto, oltre che nella directory, anche nell'header di ciascun settore al momento della formattazione. Morale della favola: se vuoi realmente cambiare l'ID devi riformattare il disco.

```
  100 REM DIMOSTRATIVO
  110 REM CAMBIO DISKNAME
  120 REM PER QUALSIASI COMMODORE
  130 REM BY M. DELL'ABATE
  140 :
  141 PRINTCHR$(147)"INSERISCI IL
       DISCO E PREMI UN TASTO"
  142 GET A$:IF A$="" THEN 142
  145 PRINT"NOME ATTUALE DEL DISC
      O:"
  150 X4=1:GOSUB 14000:PRINTX1$
  200 INPUT "NUOVO NOME (*=ANNULL
      O)";X1$:IF X1$="*" THEN END
  220 X4=0:GOSUB 14000:PRINT:PRIN
      TX2;X0$:PRINT
  300 :
 9999 END
14000 IF X4 THEN 14020
14005 X2=0:X1=LEN(X1$):IF X1<1 OR
       X1>16 THEN X0$="ERR":RETUR
      N
14010 X2$="":FOR X2=1 TO 16-X1:X2
      $=X2$+CHR$(160):NEXT:X1$=X1
      $+X2$
14020 CLOSE 2:CLOSE 15:OPEN 15,8,
      15,"I":OPEN 2,8,2,"#":GOSUB
       14060
14030 PRINT#15,"UA:";2;0;18;0:PRI
      NT#15,"B-P:";2;144
14040 IF X4 THEN X1$="":FOR X1=1
      TO 16:GET #2,X2$:X1$=X1$+X2
      $:NEXT:GOTO 14048
14045 PRINT#2,X1$;
14048 PRINT#15,"UB:";2;0;18;0:GOS
      UB 14060:PRINT#15,"I"
14050 CLOSE 2:CLOSE 15:RETURN
14060 INPUT#15,X2,X0$:RETURN
14090 REM X1$= NOME DEL DISCO
14092 REM X4<>0: LEGGE NOME DA DI
      SCO
14093 REM X4=0: SCRIVE NOME SU DI
      SCO
14094 REM X2=NUMERO DELL'ERRORE (
      0=OK)
14095 REM X0$=DESCRIZIONE DELL'ER
      RORE
14099 REM NOME: CAMBIO DISKNAME
```

14100 Finestre di testo

(Per C/64 e Vic 20)

Mediante questo sottoprogramma si possono creare vere e proprie (o quasi...) window sullo schermo.

Una window (finestra) è nel nostro caso un rettangolo, contornato da caratteri semigrafici, nel quale viene visualizzato un insieme di caratteri (parole o frasi), assegnato alla stringa X1$ prima dell'accesso alla GOSUB.

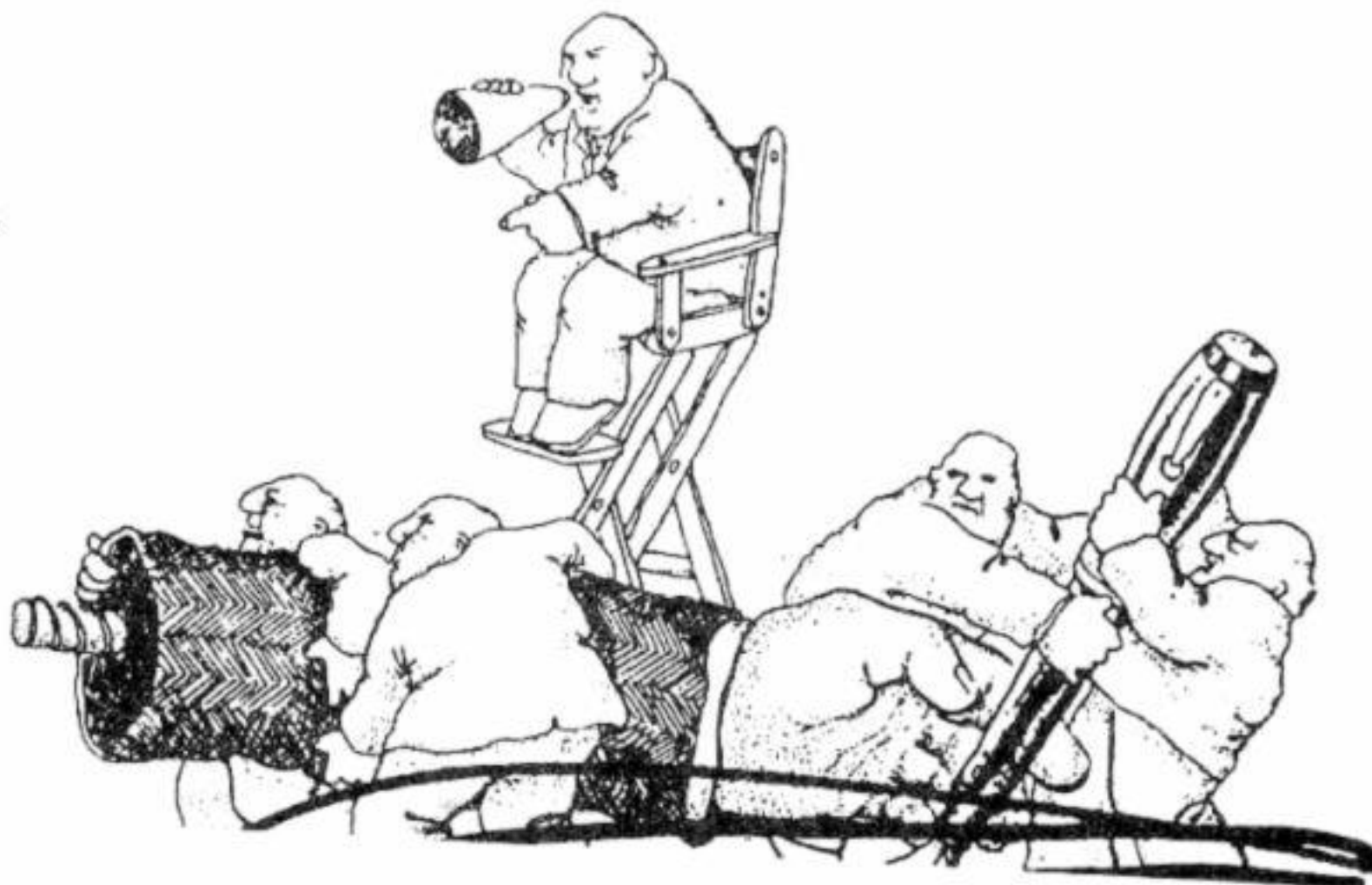
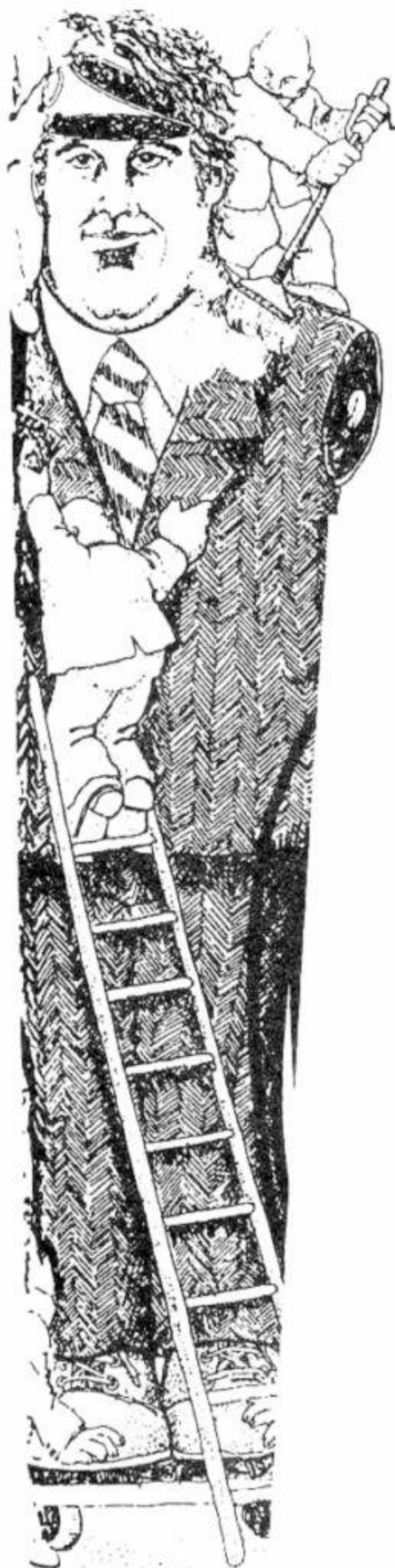
E' necessario assegnare le coordinate del vertice in alto a sinistra del rettangolo (X1, Y1) e la sua larghezza (X2): a tutto il resto penserà la routine. Si noti che le coordinate si riferiscono al primo carattere del testo; essendo la window circondata da una cornice, X1 ed Y1 non potranno assumere valori inferiori ad uno. X2 non è altro che il numero massimo di caratteri su ogni rigo. Il demo in "testa" alla routine descrive il suo funzionamento megio di una qualsiasi spiegazione sulla rivista.

Assegnando valori inadeguati alle variabili sopra descritte, X0$, come sempre, conterrà "ERR".

N.B. I possessori di Vic 20 devono fare i conti con le ridotte dimensioni dello schermo!

```
100 REM ESEMPIO D'USO
110 REM FINESIRE DI TESTO
120 REM PER C/64 E VIC 20
130 :
140 PRINTCHR$(147):REM PULISCE
    SCHERMO
150 X1$="QUESTO E' UN ESEMPIO D
    I FINESTRA DI TESTO. E' POS
    SIBILE CREARNE"
160 X1$=X1$+" DI TUTTE LE DIMEN
    SIONI..."
170 X1=10:Y1=10:X2=10:GOSUB 141
    00
180 FOR I=0 TO 1999:NEXT
190 X1$="QUESTA FINESTRA E' PIU
    ' LARGA. LE FINESTRE POSSON
    O SOVRAPPORSI"
200 X1$=X1$+" E CREARE COSI' EF
    FETTI INTERESSANTI... ASPET
    TA, PER FAVORE..."
210 X1=12:Y1=3:X2=20:GOSUB 1410
    0
220 FOR I=0 TO 2999:NEXT
230 X1$="CON UN PO' DI PA-ZIENZ
    A E DI ABI- LITA' SI PUO' F
    A-RE IN MODO"
240 X1$=X1$+" CHE LEPAROLE NON
    VEN-  GANO TAGLIATE A  FINE
    RIGO (A TAL PROPO"
250 X1$=X1$+"SITO VEDI ILLISTAT
    O DI QUESTOPROGRAMMA)."
260 X1=21:Y1=10:X2=17:GOSUB 141
    00
270 FOR I=0 TO 1999:NEXT
280 X1$="PREMI UN TASTO"
290 X1=3:Y1=3:X2=1:GOSUB 14100
300 POKE 198,0:WAIT 198,1:POKE
    198,0
310 PRINTCHR$(147):PRINT"GUARDA
     QUESTO EFFETTO...":FOR I=0
     TO 999:NEXT
320 X1$="":FOR I=0 TO 100:X1$=X
    1$+CHR$(32):NEXT:X2=9
330 FOR B=1 TO 6:X1=B:Y1=B:GOSU
    B 14100:NEXT
400 :
9999 END
14100 X9=LEN(X1$):X0$="OK"
14105 IF X1<1 OR X1>37 OR Y1<1 OR
       Y1>24 OR X2<1 OR X2>40 THE
      N X0$="ERR":RETURN
14110 POKE 211,X1-1:POKE 214,Y1-1
      :SYS58640
14111 PRINT"╭";:FOR Y9=1 TO X2:PR
      INT"─";:NEXT:PRINT"╮"
14115 FOR Y9=1 TO X9 STEP X2
14120 X2$=MID$(X1$,Y9,X2):PRINT T
      AB(X1-1);"|";X2$;:Y6=X2-LEN
      (X2$)
14121 IF Y6 THEN FOR Y5=1 TO Y6:P
      RINTCHR$(32);:NEXT
14122 PRINT TAB(X1+X2);"|"
14125 NEXT
14126 PRINT TAB(X1-1);"╰";:FOR Y9
      =1 TO X2:PRINT"─";:NEXT:PRI
      NT"╯"
14130 RETURN
14190 REM X1/Y1=COORDINATE DI PAR
      TENZA
14192 REM (CORNICE ESCLUSA).
14193 REM X2=LARGHEZZA (NUMERO MA
      SSIMO
14194 REM DI CARATTERI PER RIGO).
14195 REM X1$=STRINGA DA STAMPARE
14199 REM NOME: FINESTRE DI TESTO
```

Maurizio Dell'Abate

NUOVO SISTEMA

Enciclopedia delle routine in Linguaggio Macchina

a cura di Alessandro de Simone

Sul N.31 di Commodore Computer Club è iniziata una nuova rubrica che ha lo scopo di venire incontro ai principianti (senza trascurare gli esperti), che desiderano potenziare al massimo le caratteristiche del proprio computer.

Il Basic presenta, infatti, carenze notevoli che possono esser limitate ricorrendo all'uso di routine in linguaggio macchina (LM): è sufficiente attenersi alle istruzioni pubblicate per utilizzare i sottoprogrammi LM con la massima semplicità. Gli "esperti" potranno fare a meno di seguire le istruzioni ed utilizzare direttamente i programmi L.M. pubblicati.

I principianti, invece, è opportuno che leggano con attenzione le "istruzioni per l'uso".

Come utilizzare le routine

0/ Se questa è la prima volta che leggete la rivista, accendete il vostro Commodore 64 e saltate al punto N.2.

1/ Accendete il computer e, se desiderate "fondere" alcune (o tutte) le routine di questo numero con quelle tratte dai numeri precedenti (a patto, ovviamente che ne siate in possesso), caricate il file-programma "Nuovo Sistema" (nome standard adottato) con una delle due forme sintattiche che si riferiscono, rispettivamente, ai possessori di nastro o disco:

Load "Nuovo Sistema",1,1
Load "Nuovo Sistema",8,1

Subito dopo digitate NEW e premete il tasto Return.

2/ Caricate il programma "Fissa Top di memoria" e lanciatelo col solito RUN. Alla domanda "Ultima locazione?" digitate 20000 e, alla successiva richiesta di conferma, premete il tasto "S". Le altre informazioni che appaiono sul video possono esser comprese solo dagli esperti: i principianti possono tranquillamente ignorarle e saltare alla prossima fase (N.3).

3/ Caricate (o digitate dalla rivista) il programma "Caricatore".

4/ Digitate dalla rivista la routine che interessa (scritta sempre in Basic, contenente in prevalenza istruzioni Data e numerata da 1000 in poi).

5/ Effettuate una copia di sicurezza del programma che rappresenta la "fusione" dei due listati ("Caricatore" + routine Basic pubblicata)

6/ Dopo aver digitato Run, alla domanda "Da quale locazione?" rispondete con l'indirizzo iniziale suggerito nello stesso titolo della routine in oggetto. Se il computer, dopo alcuni secondi, visualizza, come indirizzo finale, un valore diverso da quello pubblicato nel titolo (oppure il messaggio "Errore di trascrizione"), interrompete il lavoro (tasti Run/stop e Restore) e verificate con attenzione quanto avete trascritto da rivista.

Se, invece, compaiono messaggi "confortanti" (Routine allocata da... a... Attivare con Sys... ed altre informazioni comprensibili dagli esperti), digitate il programma dimostrativo e lanciatelo: da questo momento avete a disposizione una nuova routine LM da attivare mediante SYS come indicato nelle istruzioni pubblicate per ciascuna routine.

7/ Ripetete le operazioni, dal punto 3 in poi, per ciascuna routine pubblicata che intendete "collezionare" non dimenticando di digitare NEW dopo ogni felice conclusione della fase N.6. Ai principianti consigliamo vivamente di trascriverle tutte in modo da aumentare la propria esperienza e, soprattutto, per evitare incomprensioni degli articoli che leggeranno su Commodore Computer Club.

8/ Caricate, dopo un nuovo NEW, il programma "Save Zona Ram" e, dopo il Run, alla domanda "Da quale locazione?" rispondete con 20000. Alla seconda domanda "A quale locazione?" ripondete digitando l'indirizzo finale dell'ultima routine trascritta. A seconda se avete un registratore oppure un drive, sul nastro (oppure sul disco) vi ritroverete, dopo aver risposto alle varie domande, il file-programma "Nuovo Sistema" (nome che suggeriamo di assegnare quando compare la relativa domanda). Tale file-programma (da caricare come indicato al punto 1) sarà utilissimo sia per arricchire la vostra raccolta (trascrivendo le routine dei prossimi numeri di Commodore Computer Club), sia per utilizzarle in vostri listati.

9/ Digitate SYS 64738 oppure premete il tasto di Reset (se lo possedete) in modo da rimettere "a posto" il computer. Caricate il programma "Fissa Top di memoria" e rispondete con 20000 alla domanda che vi porrà: da questo momento potrete disporre sia delle consuete istruzioni Basic che delle routine LM richiamabili con le corrispondenti SYS. Non dimenticate la presene fase (N. 9) tutte le volte che premete il tasto Reset o dopo un reset software (SYS 64738). Se, invece, spegnete il computer, sarà necessario attuare la frase N. 1 e N. 2 per inserire nuovamente nel calcolatore le nuove routine ed usarle senza pericolo.

Collaborazione dei lettori

I lettori che intendono collaborare devono inviare (almeno) tre routine, relativi listati dimostrativi ed articoli esplicativi. Le norme da seguire per la stesura dei listati (piuttosto rigide, per ovvi motivi di compatibilità) sono state segnalate sul N.31. Per ulteriori informazioni, comunque, è possibile telefonare in Redazione (02/8467348) chiedendo di Michele Maggi.

(Le routine di questo numero sono opera di Giancarlo Mariani)

```
100 PRINTCHR$(147)"FISSA TOP DI
    MEMORIA"
110 INPUT "ULTIMA LOCAZIONE";X:
    X=X-1
112 PRINT:PRINT"I VALORI ATTUAL
    I SONO:":PRINT
113 X1=INT(X/256):X2=X-(X1*256)
115 PRINT"PEEK(55):"PEEK(55):PR
    INT"PEEK(56):"PEEK(56)
116 PRINT"FRE(0):"FRE(0)
117 PRINT:PRINT"I VALORI NUOVI
    SAREBBERO:":PRINT:PRINT"PEE
    K(55):"X2
118 PRINT"PEEK(56):"X1
120 PRINT:PRINT"CONFERMI? (S/N)
    "
130 IF PEEK(197)=64 THEN 130
135 IF PEEK(197)<>13 THEN POKE
    198,0: RUN
150 POKE 55,X2:POKE 56,X1: RUN1
    60
160 PRINT"FRE(0):"FRE(0):PRINT:
    PRINT"NEW"
```

```
100 REM PROGRAMMA CARICATORE
110 REM PER ENCICLOPEDIA
120 REM DI ROUTINE IN
130 REM LINGUAGGIO MACCHINA
140 :
150 REM PER UTILIZZARLO, LEGGI
    LE ISTRUZIONI PUBBLICATE SU
160 REM COMMODORE COMPUTER CLUB
170 :
180 Y=-1:GOSUB 1000:PRINT:INPUT
    "DA QUALE LOCAZIONE";X
190 READ W:Y=Y+1:IF W<0 THEN 21
    0
200 GOTO 190
210 PRINT"PRIMA LOCAZIONE ="X
220 PRINT"ULTIMA LOCAZIONE ="X+
    Y-1:PRINT
230 PRINT"CONFERMI? (S/N)"
240 GET A$:IF A$="" THEN 240
250 IF A$="S" THEN RESTORE :T=X
    :GOTO 280
260  RUN
270 :
```

```
280 GOSUB 1000:PRINT:PRINT"ATTE
    NDERE...":PRINT:W=0
290 READ B:IF B>=0 THEN POKE T,
    B:T=T+1:W=W+B:GOTO 290
300 READ B:IF B<>W THEN PRINT:P
    RINTCHR$(18)"ERRORE DI IRAS
    CRIZIONE":END
310 PRINT"ROUTINE ALLOCATA DA"X
    "A"T-1"COMPR.":PRINT
320 PRINT"ATTIVARE CON SYS"X:PR
    INT
330 X1=INT(X/256):X2=X-(X1*256)
    :PRINT"POKE44,"X1":POKE43,"
    X2;
340 Y1=INT(T/256):Y2=T-(Y1*256)
    :PRINT":POKE46,"Y1":POKE45,
    "Y2:END
```

```
100 REM PROGRAMMA IDONEO PER
110 REM REGISTRARE ZONE DI
120 REM MEMORIA AUTOMATICAMENTE
130 REM SU CASSETTA O DISCO
140 :
150 PRINTCHR$(147):INPUT "LOCAZ
    IONE INIZIALE";X
160 INPUT "LOCAZIONE FINALE";T
170 PRINT:PRINT"INIZIO:"X:PRINT
    "FINE:"T
180 IF T<=X THEN  RUN
190 PRINT:PRINT"CONFERMI? (S/N)
    "
200 GET A$:IF A$="" THEN 200
210 IF A$<>"S" THEN  RUN
220 PRINT:INPUT "NOME FILE";A$
230 PRINT"1- CASSETTA":PRINT"2-
     DISCO"
240 GET B$:IF B$="" THEN 240
250 IF B$="1" THEN W=1:GOTO 280
260 IF B$="2" THEN W=8:GOTO 280
270 GOTO 240
280 PRINTCHR$(147);
290 POKE 198,5:POKE 631,19:POKE
     632,13:POKE 633,13:POKE 63
    4,13:POKE 635,0
300 X1=INT(X/256):X2=X-(X1*256)
    :PRINT"P┌44,"X1":P┌43,"X2;
310 Y1=INT(T/256):Y2=T-(Y1*256)
    :PRINT":P┌46,"Y1":P┌45,"Y2
320 PRINT:PRINT:PRINT"S♠"CHR$(3
    4)A$CHR$(34)","W",1"
330 PRINT:PRINT:PRINT:PRINT:PRI
    NT"SYS64738"
```

Salti... etichettati

Gosub Label (20303/20445)

Goto Label (20446/20562)

Talvolta mi è capitato, sfogliando manuali di alcuni computer, di notare istruzioni Basic piuttosto strane, come Goto e Gosub non seguiti da un numero di linea ma da una stringa alfanumerica.

Osservando gli esempi proposti si notava che la stessa stringa si trovava anche in altro punto del programma.

Come facilmente intuibile, tali istruzioni trasferivano il controllo del programma alla linea che conteneva la cosiddetta "etichetta" specificata nella chiamata.

Queste istruzioni hanno il medesimo effetto delle normali Goto/gosub ma hanno l'innegabile vantaggio di consentire una più facile leggibilità del programma e, soprattutto, l'"inattaccabilità" dello stesso da parte di cambiamenti di numeri di linea.

Pensate, infatti, a come sia molto più significativa un'istruzione del tipo Gosub"SOMMA" piuttosto che un'"asettica" Gosub300. Oltretutto, in caso di rinumerazione del programma (o di trasferimento della routine in altro luogo) non sarà necessario cambiare anche tutti i salti come succedeva, al contrario, con le Goto e le Gosub tradizionali.

Come unico svantaggio si ha una non facile reperibilità delle subroutine "etichettate" (infatti non si puo' scrivere un'istruzione del tipo List"SOMMA") specialmente in programmi molto lunghi. Questo inconveniente è comunque ovviabilissimo tramite una routine di ricerca che sarà pubblicata prossimamente in questa stessa rubrica.

Sembra dunque che risulti interessante usufruire di istruzioni di tal tipo nei propri programmi ma, purtroppo, non sono presenti nel Basic Commodore, che risponde ostinatamente "UNDEF'D STATEMENT" ad ogni tentativo di uso.

A venirci incontro è ancora una volta l'eccezionale sistema operativo (SO) del C'64, che provvede a fornire quasi tutti i mezzi necessari per la realizzazione del progetto. Spulciando pagine su pagine di disassemblato si scoprono infatti utili routine atte alla ricerca di una linea Basic in memoria, alla gestione degli errori, eccetera.

I comandi da impartire per realizzare un salto etichettato sono i seguenti:

1' Inizia a cercare l'etichetta dalla prima (o dall'ultima) linea di programma.
2' Nel caso di etichetta trovata, confrontala con quella impostata.
3' Se il confronto ha esito positivo, calcola il salto ed effettualo...
4' ...altrimenti passa ad una nuova linea e ricomincia dal punto 2. Ripeti fino alla fine del programma.
5' Nel caso di etichetta non trovata, segnala un errore.

I punti 1 e 3 sono svolti quasi interamente dal S.O; i punti 4 e 5 a... metà, quindi per il programmatore resta solo il compito del confronto delle etichette ed altri, quali la gestione di alcuni puntatori, il controllo della fine del programma e simili.

Le routine proposte (separate, una per il Goto e l'altra per la Gosub) sono scritte inte-

ramente in LM, per ovvi motivi di velocità di esecuzione e sono liberamente rilocabili in qualsiasi area di memoria RAM (tralasciando ovviamente quelle "pericolose" come pagina 0, memoria di schermo, eccetera) in modo da permettere all'operatore la massima flessibilità di utilizzo, soprattutto insieme ad altri programmi.

Consigliamo ai principianti, tuttavia, di allocarle da 20303 (Gosub Label) e da 20446 (Goto Label) non dimenticando di seguire i suggerimenti riportati nel riquadro di queste pagine.

Non abbiamo trattato, finora, il problema di come "etichettare" le routine all'interno di un programma. Ovviamente non possiamo scrivere una linea del tipo

100 SOMMA

perchè l'interprete, incontrandola, segnalerebbe un Syntax error, arrestando l'esecuzione del programma. Bisogna quindi cercare un modo per fornire le etichette al programma per non avere "effetti collaterali" durante l'esecuzione del programma stesso.

Analizzando alcune possibilità si scopre che, per lo scopo prefissato, la REM è l'unica istruzione che "si presenta proprio bene". REM, come ben si sa, è utilizzata soltanto per inserire commenti in un listato (per migliorarne la leggibilità) e non ha effetti pratici sullo svolgimento dello stesso.

Nel nostro caso, invece, la rem ha la possibilità di "elevarsi di grado" permettendo anche al Basic V2.0 di raggiungere (o quasi) le vette di macchine ben più potenti e costose. La sintassi che gestisce queste nuove "istruzioni" è la seguente:

SYS XXXX,"LABEL"

In cui "XXXX" è l'indirizzo di partenza della routine (ovviamente diverso, come suggerito prima, per la Goto 20446 e la Gosub 20303). "LABEL", invece, è appunto il nome della routine che si desidera venga eseguita. Questa puo' essere una stringa scritta in forma esplicita oppure un'espressione contorta quanto si vuole purchè di tipo stringa.

In testa alla routine da chiamare va messo ovviamente il nome corrispondente nella forma:

10000 REM*LABEL

In pratica dopo il numero di linea (10000 nell'esempio) andrà scritta la parola chiave REM, quindi, SENZA LASCIARE ALCUNO SPAZIO, un asterisco e, subito dopo, l'etichetta che puo' essere lunga quanto si vuole, compatibilmente con la massima lunghezza della linea Basic.

Nelle linee successive andrà scritta la subroutine chiamata, che dovrà terminare con l'istruzione Return in caso di chiamata con la Gosub Label.

L'asterisco è indispensabile, pena il non riconoscimento dell'etichetta, ed è stato messo apposta per differenziare le REM "etichettate" dalle altre. Questo carattere puo' comunque essere cambiato a piacere dai programmatori più esperti disassemblando la routine e modificando l'opportuna istruzione.

Se un'etichetta non viene trovata dopo aver esaminato tutto il programma viene segnalato un "UNDEF'D STATEMENT" e viene arrestata l'esecuzione.

Nel digitare i due programmi bisogna fare molta attenzione a copiare le linee di Data e salvare il programma appena digitato PRIMA di farlo eseguire. Questo eviterà imprecazioni ed inutili perdite di tempo in caso di errori di battitura.

Compito dei lettori è quello di migliorare le routine, magari aggiungendovi anche quelle di Goto e Gosub calcolato proposte in precedenza su CCC.

```
1000 PRINTCHR$(147)"GOSUB LABEL.
     ESEGUIRE SYS(ALLOCAZIONE)"
1005 PRINT"SEGUITO DALLA STRINGA
     . ESEMPIO:"
1006 PRINT"SYS XXXX,A$":PRINT"SY
     S XXXX,"PIPPO"
1007 RETURN
1100 DATA 032,253,174,032,158
1110 DATA 173,032,130,183,192
1120 DATA 000,240,087,132,002
1130 DATA 169,000,133,020,133
1140 DATA 021,024,032,019,166
1150 DATA 008,176,004,177,095
1160 DATA 240,068,160,002,177
1170 DATA 095,133,020,200,177
1180 DATA 095,133,021,040,144
1190 DATA 231,230,020,208,002
1200 DATA 230,021,200,177,095
1210 DATA 201,143,208,218,200
1220 DATA 177,095,201,042,208
1230 DATA 211,200,132,253,165
1240 DATA 095,024,101,253,133
1250 DATA 095,144,002,230,096
1260 DATA 160,255,200,177,095
1270 DATA 240,006,209,034,240
1280 DATA 247,208,184,196,002
1290 DATA 240,008,024,144,177
1300 DATA 162,017,108,000,003
1310 DATA 198,020,166,020,232
1320 DATA 208,002,198,021,169
1330 DATA 003,032,251,163,165
1340 DATA 123,072,165,122,072
1350 DATA 165,058,072,165,057
1360 DATA 072,169,141,072,032
1370 DATA 121,000,032,163,168
1380 DATA 076,174,167,-1,17585
```

```
10 REM DIMOSTRATIVO DI GOSUB L
   ABEL
15 :
17 :
20 X=20303:REM  INDIRIZZO SUGG
   ERITO SU COMMODORE COMPUTER
    CLUB
25 :
40 SYSX,"ROUTINE DI PROVA"
50  RUN
99 :
```

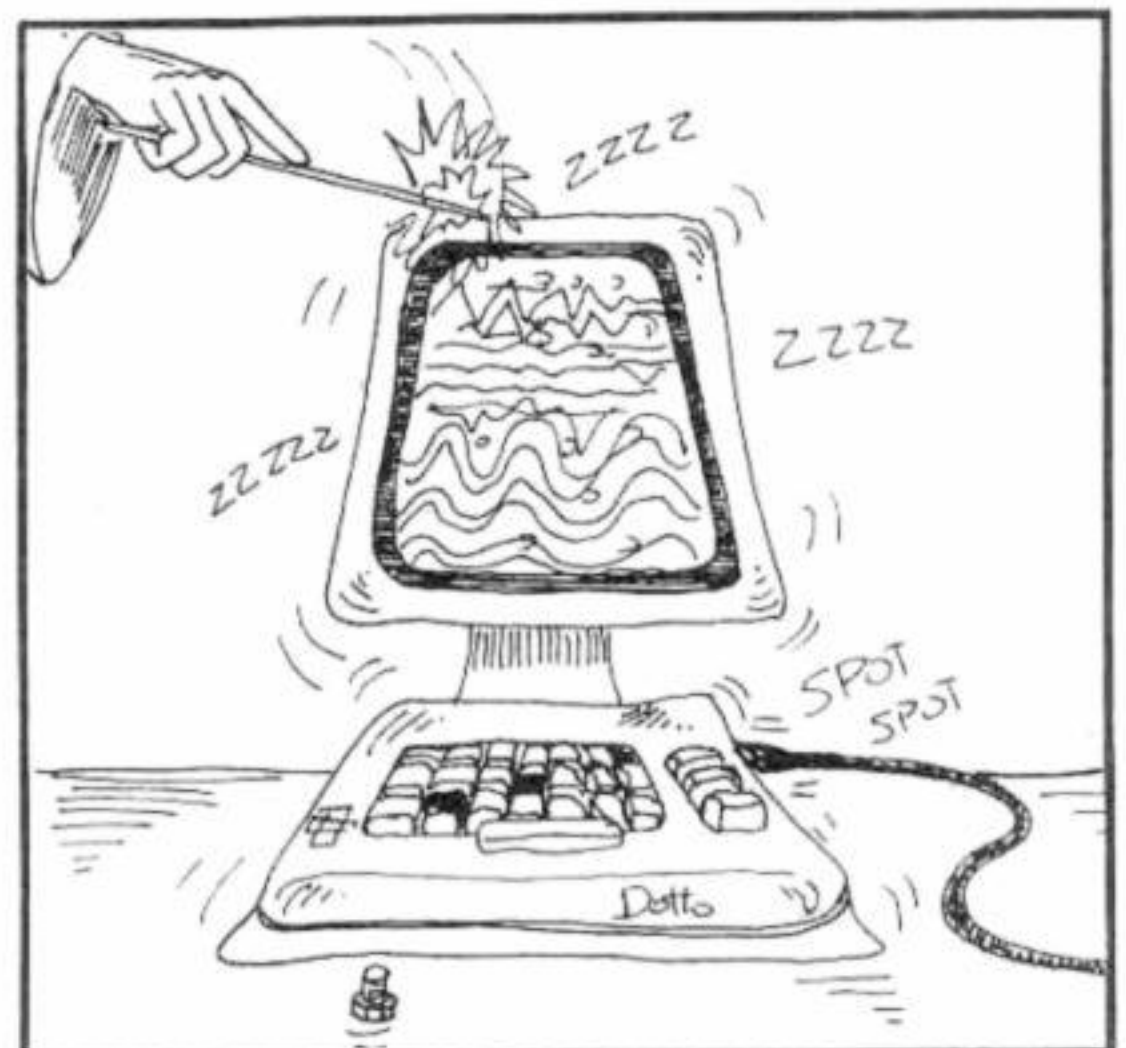
```
100 REM * ROUTINE DI PROVA
105 INPUT "ROUTINE (ES. CIAO O
    PROVA)";A$
110 SYSX,A$:RETURN
120 :
130 REM *CIAO
135 PRINT"HO CHIAMATO LA ROUTIN
    E 'CIAO'":RETURN
140 :
150 REM *PROVA
155 PRINT"HO CHIAMATO LA ROUTIN
    E 'PROVA'":RETURN
```

```
1000 PRINTCHR$(147)"GO TO LABEL:
      ESEMPIO D'USO:"
1002 PRINT"SYS XXXX,A$":PRINT"SY
     S XXXX,"PIPPO"
1003 RETURN
1100 DATA 032,253,174,032,158
1110 DATA 173,032,130,183,192
1120 DATA 000,240,087,132,002
1130 DATA 169,000,133,020,133
1140 DATA 021,024,032,019,166
1150 DATA 008,176,004,177,095
1160 DATA 240,068,160,002,177
1170 DATA 095,133,020,200,177
1180 DATA 095,133,021,040,144
1190 DATA 231,230,020,208,002
1200 DATA 230,021,200,177,095
1210 DATA 201,143,208,218,200
1220 DATA 177,095,201,042,208
1230 DATA 211,200,132,253,165
1240 DATA 095,024,101,253,133
1250 DATA 095,144,002,230,096
1260 DATA 160,255,200,177,095
1270 DATA 240,006,209,034,240
1280 DATA 247,208,184,196,002
1290 DATA 240,008,024,144,177
1300 DATA 162,017,108,000,003
1310 DATA 198,020,166,020,232
1320 DATA 208,002,198,021,076
1330 DATA 163,168,-1,14751
```

```
10 REM DIMOSTRATIVO DI GOTO LA
   BEL
17 :
20 X=20446:REM INDIRIZZO SUGGE
   RITO SU COMMODORE COMPUTER
   CLUB
25 :
40 SYSX,"ROUTINE DI PROVA"
45 :
50 REM*RIPARTI
60  RUN
90 :
100 REM*ROUTINE DI PROVA
105 INPUT "ROUTINE (ES. CIAO O
    PROVA)";A$
110 SYSX,A$
115 :
130 REM*CIAO
135 PRINT"HO CHIAMATO LA ROUTIN
    E 'CIAO'":SYSX,"RIPARTI"
140 :
150 REM*PROVA
155 PRINT"HO CHIAMATO LA ROUTIN
    E 'PROVA'":SYSX,"RIPARTI"
```

Mappa della memoria di Nuovo Sistema (elenco delle routine pubblicate)

Il primo valore indica l'indirizzo di partenza (coincidente con la SYS), mentre il secondo indica l'ultima locazione contenente l'ultimo dato della stessa routine.

20000/20011 Goto calcolato
20012/20049 Gosub calcolato
20050/20128 Interp A$
20129/20188 Cambia colore al carattere
20189/20245 Scroll carattere
20246/20302 Cancella carattere
20303/20445 Gosub Label
20446/20562 Goto Label

Disegnare col computer

Utilizziamo il video come lavagna e il computer come... gesso!

Di Michele Maggi

Recentemente (CCC N.28) ci siamo occupati dell'utilizzo dell'istruzione BOX per un'interessante e utile applicazione. Si trattava, infatti, di un programmino che generava istogrammi idonei per la rappresentazione grafica di tabelle di valori.

Oltre all'istruzione BOX abbiamo visto, anche se in modo non molto approfondito, l'istruzione DRAW che serviva, in quel programma, per il tracciamento degli assi cartesiani. L'istruzione DRAW, cuore del presente articolo, serve sia per tracciare linee che singoli punti e, come l'istruzione BOX, può essere usata per evidenziare e per cancellare.

Si ricorda che lo spigolo del video in alto a sinistra ha le coordinate X=0 e Y=0; quello in alto a destra ha: X=319 e Y=0; quello in basso a sinistra ha: X=0, Y=199. Il formato tipico (sintassi) dell'istruzione DRAW e il seguente:

DRAW 1,X,Y

In questo caso viene visualizzato un punto in alta risoluzione alle coordinate specificate da X e Y. Desiderando, invece, tracciare un'intera linea, dovremo aggiungere altre due coordinate da considerare come coordinate di arrivo.

Esempio:
DRAW 1,X,Y TO X1,Y1

Notiamo quindi che X e Y vengono considerate, dal C/16 e C/128, come le coordinate di "partenza" e che X1 e Y1 come coordinate di "arrivo".

Si intuisce che con poche istruzioni si e in grado di disegnare forme e figure di qualsiasi genere.
Variando di poco la sintassi...

DRAW 1,10,10 TO 10,80 TO 100,80 TO 10,10

...otterremo un triangolo.

Da ciò si capisce che non c'è limite alle coordinate che possiamo aggiungere a DRAW.

Esempio:

DRAW 1,X,Y TO X1,Y1 TO X2,Y2 TO ... Xn,Yn.

Come il comando BOX, anche DRAW si rivela estremamente utile e versatile.

Un'ultima considerazione prima di proseguire: la cifra 1 che segue l'istruzione DRAW indica al C/16 e al C/128 che intendiamo tracciare e non cancellare (in questo caso si dovrebbe usare 0) e può essere omessa purchè rimanga la virgola che funge da separatore fra l'istruzione DRAW vera e propria e le sue coordinate.

Ad esempio: DRAW,X,Y otterrà lo stesso effetto di DRAW 1,X,Y.

Prima di passare ad analizzare il programmino di queste pagine, ricordiamo che l'istruzione DRAW, come tutte le altre di carattere grafico, non sortirà alcun effetto se prima non selezioniamo il modo grafico.

I modi grafici

Per chi non avesse ancora molta dimestichezza con la materia riassumeremo brevemente ciò che si intende per "alta risoluzione" e "modo grafico".

Alta risoluzione

Normalmente quando accendiamo il C/16 (o Plus/4 o C/128), si presenta uno schermo comunemente definito in Bassa Risoluzione.

Che cosa significa?

Significa che saremo in grado di controllare, con i movimenti del cursore,

non un singolo punto ma un "blocchetto" di 8*8 punti.

Da ciò si deduce che anche usando intensivamente tutti i caratteri semigrafici a disposizione sulla tastiera non saremo mai in grado di creare disegni precisi.

A questo fatto si può ovviare ricorrendo, appunto, all'alta risoluzione.

Per alta risoluzione intendiamo uno schermo in cui il più piccolo punto controllabile e un DOT e non un intero carattere (il blocchetto prima visto) che invece è formato da una combinazione di 8*8=64 DOT che, accesi o spenti, formano il disegno del carattere.

Mentre lo schermo in bassa risoluzione può contare solo su 1000 posizioni (tante quente sono le locazioni di memoria riservate al video) lo schermo in A/R può vantare ben 64000 punti perchè, come precedentemente esposto, ogni carattere é formato da 64 Dot che, moltiplicati per il numero di caratteri visualizzabili sullo schermo (1000), forniscono come risultato 64000.

Il nostro computer può controllare ognuno di questi punti. Per ciò che concerne i modi grafici basta ricordare che sono cinque:

** GRAPHIC 1 che crea uno schermo in A/R.*
** GRAPHIC 2 che crea uno schermo in A/R ma riservando parte dello schermo (5 righe) per il modo testo.*
** GRAPHIC 3 e GRAPHIC 4 che funzionano come i precedenti ma in modo multicolor.*
** GRAPHIC 0 che riporta lo schermo in modo testo.*

Oltre al valore (da 0 a 4) relativo al tipo di A/R desiderata, si può aggiungere un "1" che determinerà la cancellazione di ciò che é eventualmente presente sullo schermo grafico selezionato.

L'istruzione GRAPHIC 1,1 ,ad esempio, "porterà" lo schermo in A/R cancellando ciò che era presente sulla pagina grafica.

Raffaello '64 è uno dei più potenti software per disegnare col computer. Prodotto dalla Systems Editoriale, richiede solo l'uso del joystick e può essere richiesto alla nostra redazione. Costa solo lire 10.000 + spese di spedizione.

Il cursore in A.R.

Mentre in bassa risoluzione siamo abituati a vedere lampeggiare il cursore, notiamo che questo non compare in A/R.

Anche in A/R, però, esiste un cursore chiamato più propriamente CP (cursor pixel) che, di norma, è invisibile ma può essere spostato mediante l'istruzione LOCATE il cui formato tipico è il seguente: LOCATE X,Y.

Chiaramente X e Y rappresentano le coordinate alle quali si vuole posizionare il CP.

Per visualizzare il CP sarà necessario ricorrere opportunamente ad alcune istruzioni grafiche, tra cui DRAW.

Nel Basic 3.5 del C/16, e nel 7.0 del C/128, esistono due funzioni che indicano le coordinate X e Y relative alla posizione del CP:

RDOT(0) che indica la coordinata X.
RDOT(1) che indica la coordinata Y.

Specificata la posizione del CP mediante LOCATE, sarà possibile tracciare una linea a partire da quel punto con DRAW 1 TO X1,Y1 perchè la posizione "attuale" del CP verrà considerata come punto di partenza. Sarà quindi possibile omettere le prime due coordinate X e Y dell'istruzione-DRAW.

Il programma

Passiamo ora all'analisi del funzionamento del breve listato. Innanzi tutto, grazie alla sua brevità, vale sicuramente la pena di digitarlo, se non altro per imparare qualcosa di nuovo.

Dato il RUN, viene visualizzato un CP lampeggiante al centro dello schermo in A/R (coordinate 160,100).

Nell'angolo superiore sinistro del video appare la scritta MUOVI. Ciò significa che, agendo sui tasti cursore, sarà possibile spostare il CP senza lasciar traccia.

Premendo, invece, F1 compare la scritta DISEGNA: premendo uno dei tasti cursore, il CP si sposterà lasciando una scia dietro di sè.

Premendo F2 si ritorna in modo MUOVI, mentre, con F3, si ottiene il riempimento dell'area tracciata dopo una richiesta di conferma.

Se, infatti, premiamo F3 quando il CP e all'interno di un'area chiusa, otterremmo il suo "riempimento"; se invece premiamo F3 quando il CP non è contenuto in alcun riquadro delimitato, verrà riempito tutto lo schermo e ciò giustifica la richiesta di conferma. Il tasto F8 resetta il programma facendolo ripartire dall'inizio.

Come funziona il programma

Il programma è talmente semplice che una dettagliata spiegazione sarebbe superflua; qui di seguito vengono comunque analizzati i passi più importanti.

La linea 10 stabilisce i colori, il modo grafico e inizializza alcune variabili. La 20 annulla i tasti funzione standard. Le linee 30-40 e 130 si occupano del lampeggio del CP fino a che non viene premuto un tasto. Le linee 50-80 tracciano il disegno o muovono semplicemente il CP in funzione del valore di M che viene stabilito dalle linee 100-110 in relazione al modo Disegna o Muovi (rispettivamente 1 e 0). La linea 90 realizza il Paint cioè il riempimento del disegno. La 120 fa ripartire il programma da capo se viene premuto F8.

Lista Variabili

RI= Coordinata Y
COL= Coordinata X
M= Modo di tracciamento o movimento e cancellazione

Si noti che se in modo MUOVI il CP passa al di sopra di una linea, questa verrà cancellata nel punto di contatto col CP.

Conclusioni

Il programma non è (e non pretende di essere) una utility grafica di carattere professionale, per cui non potrebbe reggere confronti di nessun genere con packages (magari in Linguaggio Macchina) dotati di prestazioni strabilianti.

Il listato si presta però a miglioramenti che sicuramente saranno effettuati dai lettori più volonterosi.

Ecco alcuni suggerimenti

* *Creare la possibilità di disegnare in modo Multicolore.*
* *Aggiungere le opzioni CIRCLE e BOX.*
* *Creare una routine di salvataggio della pagina grafica a disegno ultimato.*

Concludiamo quindi esortando il lettore a considerare il programmino come un semplice punto di partenza dal quale sviluppare programmi più complessi e completi.

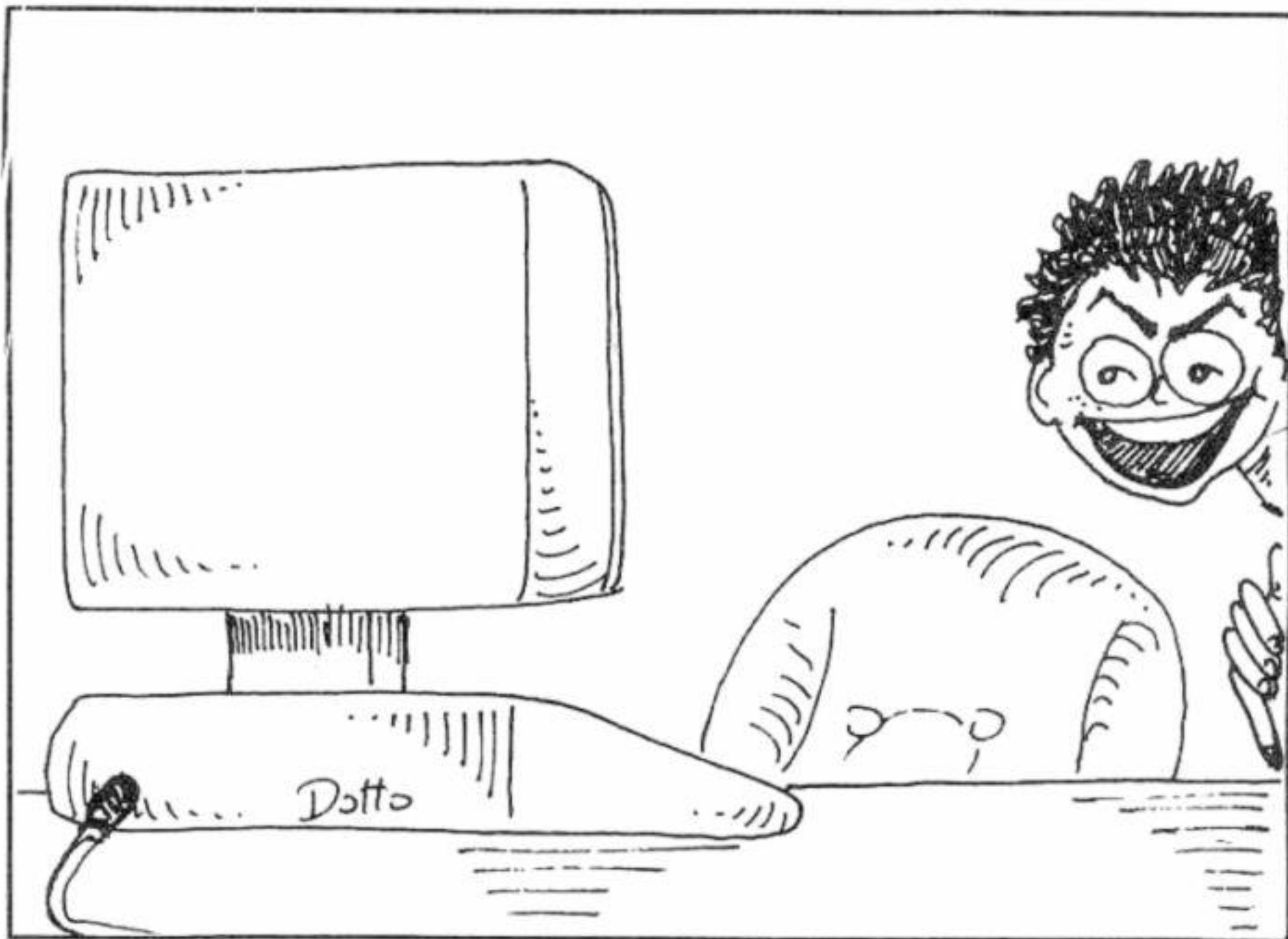
Un'ultima avvertenza per i possessori del C/16: quando si imposta uno dei modi grafici il computer riserva ben 10K byte di memoria RAM per la gestione dello schermo grafico. Sarà bene, quindi, limitarsi ad apportare brevi modifiche (= listati con poche linee Basic) in modo da evitare il tipico messaggio "Out Of Memory Error".

Esiste comunque l'istruzione GRAPHIC CLR che, disattivando lo schermo grafico, "restituisce" i 10 K al Basic.

```
1 REM DISEGNI COL C/16, PLUS/4
2 REM E C/128 IN MODO 128
4 :
5 REM BY MICHELE MAGGI
6 :
10 COLOR0,1:COLOR4,1:COLOR1,2,1:RI=100
15 COL=160:GRAPHIC1,1:CHAR,0,0,"MUOVI "
20 FORI=1TO8:KEYI,CHR$(132+I):NEXTI
30 DRAW1,COL,RI
40 GETA$:IFA$=""THEN130
50 A=ASC(A$):IFA=29THENDRAWM,COL,RI:COL=COL+1:REM CRSR RIGHT
60 IFA=157THENDRAWM,COL,RI:COL=COL-1:REM CRSR LEFT
70 IFA=145THENDRAWM,COL,RI:RI=RI-1:REM CRSR UP
80 IFA=17THENDRAWM,COL,RI:RI=RI+1:REM CRSR DWN
90 IFA=135THENGOSUB1000:IFATHENPAINT,COL+1,RI+1:REM F3
100 IFA=133THENM=1:CHAR,0,0,"DISEGNA ":REM F1
110 IFA=134THENM=0:CHAR,0,0,"MUOVI   ":REM F2
120 IFA=140THENRUN:REM F8
130 DRAW0,COL,RI:GOTO30
1000 A=0:CHAR,0,0,"SICURO? "
1010 GETA$:IFA$=""THEN1010
1020 IFA$="S"THENA=135
1025 CHAR,0,0,"         "
1030 RETURN
```

Anti decompilatore per Commodore 64

Come mandare al diavolo (via software) chi desidera esaminare i nostri listati.

di Giancarlo Mariani

Senza ripetere cose dette decine di volte, ci limitiamo a ricordare che esistono due tipi di protezioni: contro la listatura e contro la copiatura. Il presente articolo si riferisce alla prima di queste.

Le ragioni per proteggere un programma sono ormai arcinote: difesa di diritti d'autore, limitazione della diffusione di copie "pirata", semplice... gelosia.

La protezione contro la listatura può essere effettuata in vari modi, alterando puntatori del Basic, inserendo opportuni codici all'interno del programma, oppure, e arriviamo al dunque, "compilando" il programma Basic.

Che cosa vuol dire compilare?

Compilare significa, nel gergo informatico, "tradurre" un programma, trasformarlo in codici opportuni, non più listabili facilmente nè modificabili. Tale operazione di solito viene compiuta per aumentare la velocità di esecuzione del programma Basic e, spesso, anche per proteggere il programma da occhi indiscreti.

Per compilare occorre un opportuno programma chiamato, appunto, "Compilatore".

Esistono vari tipi di compilatori, quali

compilatori Pascal, Fortran, C, Basic e altri.

A noi, ovviamente, interessano i compilatori utilizzabili col Commodore 64, piuttosto diffusi e facilmente utilizzabili. Purtroppo quasi tutti richiedono il disk drive per funzionare, pur se "generano" file programmi che possono esser registrati e caricati da nastro.

Facendo "passare" un programma Basic attraverso un compilatore, il listato originale (sorgente) viene "tradotto" in un linguaggio (oggetto) intermedio tra il Basic e il LM che ha il vantaggio di essere più comprensibile dal microprocessore che non il Basic. Da ciò segue che da un programma compilato si riesce ad ottenere una velocità di esecuzione superiore a quella del corrispondente programma originale scritto in Basic.

Un secondo, non trascurabile, vantaggio consiste nella completa illeggibilità del programma stesso da parte di terzi, ed è proprio questo l'aspetto che più interessa in questa sede.

Dato che, come detto prima, un programma compilato non risulta più listabile, sembra che, almeno in teoria, il problema sia risolto.

A metterci i bastoni tra le ruote intervengono i programmi chiamati "Decompilatori" che, lo dice la parola stessa, svolgono la funzione diametralmente opposta a quella di un compilatore. Se quindi il nostro ipotetico "pirata" possiede, oltre al nostro programma compilato, anche il relativo decompilatore, tutti i nostri sforzi per applicare una protezione risulterebbero vani.

L'uovo di Colombo

E' quindi necessario trovare un sistema per neutralizzare l'azione di un decompilatore. Una soluzione è quella di codificare ulteriormente il programma oggetto compilato in modo da renderlo incomprensibile per il decompilatore.

Il problema che ne deriva però è che, oltre a non essere più leggibile per il decompilatore, il programma codificato risulta incomprensibile anche per lo stesso

computer (!). Oltre alla codifica sarà quindi necessaria la relativa decodifica da effettuare prima della partenza del programma.

Il minilistato proposto in questo articolo svolge il compito descritto: codifica il programma compilato e aggiunge allo stesso un'opportuna routine di decodifica, che sarà chiamata automaticamente al momento del RUN.

Il programma così trattato non differisce minimamente da come era prima della stessa codifica, col risultato che se si tentasse di decompilarlo "normalmente", il decompilatore segnalerebbe errori contribuendo a scoraggiare il tentativo di risalire al listato Basic originale.

Il metodo descritto funziona solo con compilatore "BLITZ" e non con altri tipi di compilatori (Petspeed, Austro compiler, eccetera). Abbiamo scelto il Blitz perchè, oltre ad offrire una buona velocità di esecuzione, presenta i vantaggi di compilare tutte le istruzioni Basic e di far occupare al programma compilato una modesta quantità di memoria. Blitz, inoltre, è un compilatore molto diffuso tra gli utenti del C/64.

Come utilizzare la protezione

Vediamo, passo per passo, come si usa l'antidecompilatore, supponendo che il programma compilato con Blitz sia già presente su cassetta o su disco:

- *Digitare la routine proposta (o caricarla da cassetta o da disco).*
- *Dare il "RUN" alla routine.*
- *Caricare il programma compilato da proteggere.*
- *Digitare "SYS 49192" e premere il tasto Return.*
- *Dare il comando "CLR".*
- *Salvare su cassetta (oppure disco) il programma.*

Da questo momento la versione del programma registrato, pur se perfettamente funzionante, sarà protetta da indesiderate decompilazioni (provare per credere).

Esiste anche un altro vantaggio non immediatamente rilevabile: il programma codificato con la routine proposta non conterrà più eventuali messaggi in formato ASCII (formato mantenuto invece durante la fase di compilazione), rendendo molto difficoltosa la loro ricerca con programmi tipo monitor di LM o simili.

Questo fatto permette all'autore del programma Basic di inserire il proprio nome con la quasi certezza che non verrà rintracciato e, di conseguenza, modificato (vedi articolo: "Una firma indelebile nei vostri listati" N.28 di CCC).

Un consiglio da osservare è quello di non tentare di codificare programmi scritti in Basic oppure in LM poichè il listato di queste pagine funziona solo con i programmi compilati con Blitz; non si otterrebbe altro se non un "inchiodamento" della macchina o, nel migliore dei casi, un malfunzionamento del programma trattato.

Chi vuole capire come funziona il listato proposto può, servendosi di un "Monitor" LM, disassemblarlo dato che, scritto interamente in LM, è allocato a partire dall'indirizzo 49152.

A dispetto di quanto detto finora, infatti, la routine è perfettamente listabile e (soprattutto) modificabile e migliorabile dai più esperti.

```
    5 REM  ************************
       ************
    6 REM  * ANTI DECOMPILER BY M
       ARIANI G. *
    7 REM  *  SOLO PER COMPILATOR
       E BLITZ   *
    8 REM  ************************
       ************
   10 PRINTCHR$(147):PRINT" ATTEN
       DERE PREGO...":PRINT
   20 CK=0:FOR T=0 TO 109:READ A:
       POKE 49152+T,A:CK=CK+A:NEXT
   22 IF CK<>15094 THEN PRINT" ER
       RORE NEI DATI!!":END
   25 PRINT" OK."
   30 REM  *************
   40 REM  * SYS 49192 *
   50 REM  *************
10000 DATA  169,000,133,251,169,0
       09,133
10001 DATA  252,160,000,177,251,0
       73,255
10002 DATA  145,251,200,208,247,2
       30,252
10003 DATA  165,252,201,000,208,2
       39,169
10004 DATA  146,141,029,008,169,0
       12,141
10005 DATA  030,008,076,028,008,1
       69,000
10006 DATA  133,251,168,169,009,1
       33,252
10007 DATA  177,251,073,255,145,2
       51,200
10008 DATA  208,247,230,252,165,2
       52,197
10009 DATA  046,208,239,165,046,1
       41,024
10010 DATA  192,230,045,208,002,2
       30,046
10011 DATA  160,000,185,000,192,1
       45,045
10012 DATA  200,192,040,208,246,2
       34,165
10013 DATA  046,141,030,008,165,0
       45,141
10014 DATA  029,008,024,105,040,1
       44,002
10015 DATA  230,046,133,045,096
```

INVIARE TUTTA LA PAGINA ANCHE SE SI UTILIZZA UNA SOLA SCHEDA

Nome .... Cognome ....

Via .... N° .... CAP. .... Città ....

Telefono .... Orario ....

Registrate il mio abbonamento annuale a Commodore Computer Club.

☐ Ho versato oggi stesso il canone di L. 35.000 a mezzo c/c postale n° 37952207 intestato a:
Systems Editoriale - V.le Famagosta, 75 - 20142 Milano

☐ Ho inviato oggi stesso assegno bancario n..
per l'importo di L. 35.000 intestato a Systems Editoriale

Si prega di scrivere il proprio nome e l'indirizzo completo in modo chiaro e leggibile. Inviare la fotocopia del bollettino di c/c postale.

Considerando che i numeri 1, 2 e 7 sono esauriti, vogliate inviarmi i numeri arretrati .... al prezzo di L. 5.000 cadauno per richieste fino a 4 numeri, o di L. 4.000 cadauno per richieste oltre i 4 numeri arretrati, e perciò per un totale di L.............. Sono a conoscenza che i fascioli suddetti non saranno inviati in contrassegno e, pertanto, ho provveduto oggi stesso a versare il canone di L.................. a mezzo c/c postale n. 37952207 intestato a:
Systems Editoriale - V.le Famagosta, 75 - 20142 Milano

STATISTICA

Non posseggo un computer ☐

Posseggo un C64 .... si ☐ ... no ☐
Posseggo un VIC 20 .... si ☐ ... no ☐
Posseggo un Commodore Plus 14 .... si ☐ ... no ☐
Posseggo un Commodore Plus 16 .... si ☐ ... no ☐
Posseggo un registratore dedicato .... si ☐ ... no ☐
Posseggo un drive 1541 .... si ☐ ... no ☐
Posseggo una stampante .... si ☐ ... no ☐
Posseggo un monitor .... si ☐ ... no ☐

COLLABORAZIONE

A titolo di prova vi invio un articolo e la cassetta........................disco.................................
col programma che intendo proporre per la pubblicazione di cui garantisco l'originalità.

DOMANDA/RISPOSTA ....

....

....

....

....

....

RICHIESTA ARGOMENTI

Mi farebbe piacere che Commodore Computer Club parlasse più spesso dei seguenti argomenti:

1/ ....

2/ ....

3/ ....

4/ ....

GIUDIZIO SUI PROGRAMMI DI QUESTO NUMERO

Ho assegnato un voto da 0 a 10 ai programmi che indico di seguito:

A/ .... Voto ....

B/ .... Voto ....

C/ .... Voto ....

D/ .... Voto ....

PICCOLI ANNUNCI

....

CERCO/OFFRO CONSULENZA

....

INVIARE IN BUSTA CHIUSA E AFFRANCANDO SECONDO LE TARIFFE VIGENTI A:

COMMODORE COMPUTER CLUB

V.le Famagosta, 75
20142 Milano

INVIARE TUTTA LA PAGINA ANCHE SE SI UTILIZZA UNA SOLA SCHEDA

Nome .... Cognome ....

Via .... N° .... CAP. .... Città ....

Telefono .... Orario ....

Da oggi c'è un nuovo distributore di stampanti FACIT per il tuo Personal Computer IBM

brain's 505

Agenzie FACIT

Arenzano (GE) *Ing. Castellucci & C.* P.za degli Ulivi, 15 Tel.: 010/9112036 - **Bergamo** *D.I.P. Bergamo* Via Borgo Palazzo, 90 Tel.: 035/233909 - **Bologna** *D.I.P. Bologna* P.za Porta Mascarella, 7 Tel.: 051/240602 - **Castelfranco Veneto** (TV) *Vecom* Borgo Treviso, 45 Tel.: 0423/496222 **Fabriano** (AN) *D.I.P. Ancona* Via G. Tommasi, 15 Tel.: 0732/22259 - **Milano** *D.I.P. Milano* Via A. Costa, 33 Tel.: 02/2840508 2840488 - **Roma** *D.I.P. Roma* Via C. Colombo, 179 Tel.: 06/5133041 - **Torino** *Elcomin* Via Artisti, 36 Tel.: 011/832620

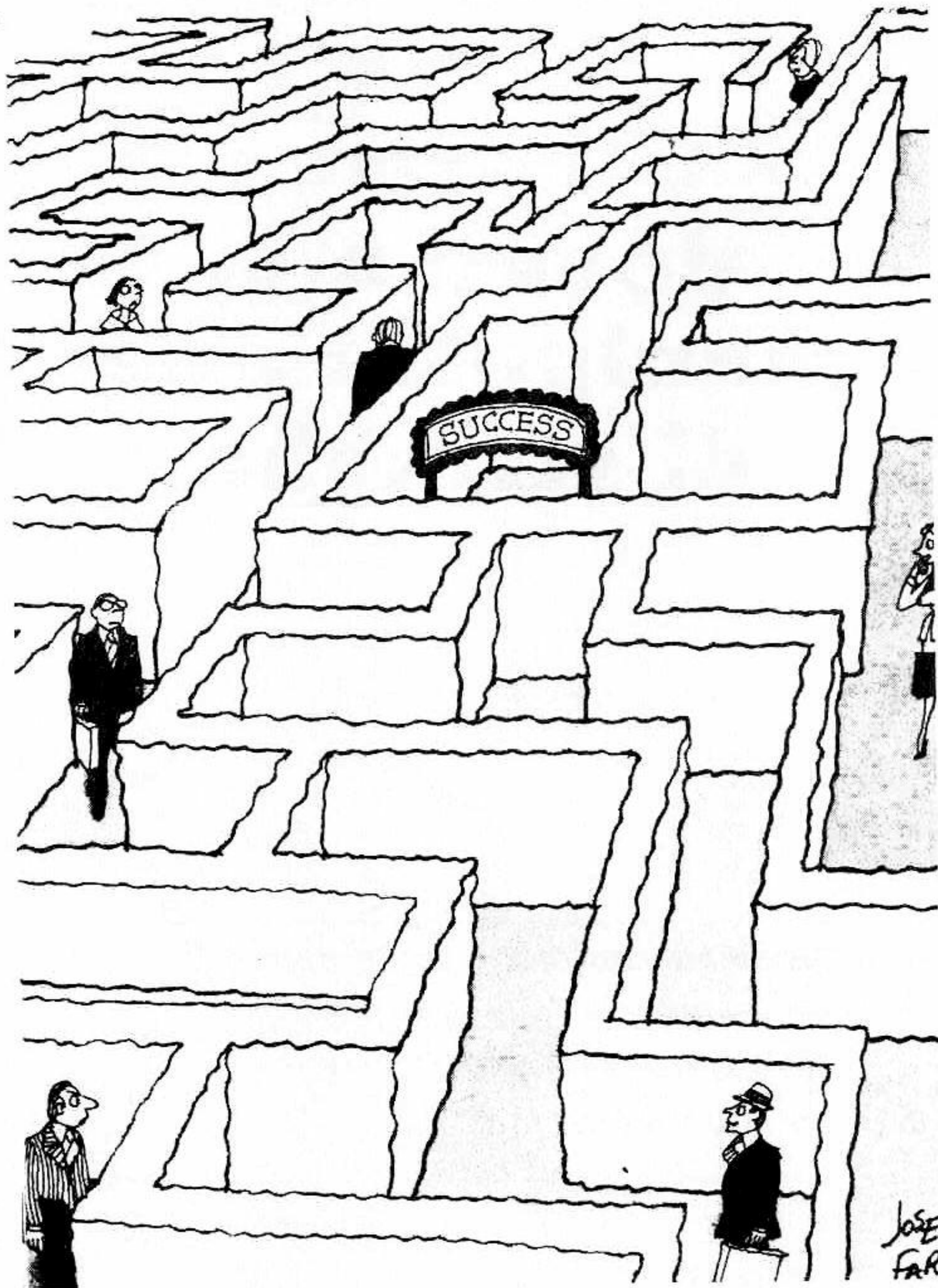
Distributori FACIT

Bassano del Grappa (VI) *Studio L.& C.* V.le Diaz, 27 Tel.: 0424/212541 - **Bassano del Grappa** (VI) *Studio L.& C.* Via Z. Bricito, 27 Tel.: 0424/29275 Sig. Luca, Sig. Venturini - **Belluno** *SCP Computer System* Via Feltre, 244 Tel.: 0437/20826 Sig. Costa Sig. Rec - **Gorizia** *Quark* Via Udine, 143 Tel.: 0481/391693 Sig. Gulin, Sig. Costa - **Mestre** (VE) *Negozio:* Via Verdi, 8/10 Tel.: 041/962866 Sig. Magnifichi - **Mestre** (VE) *Show Room:* P.za Basche, 45 Tel.: 041/958007 Sig. Magnifichi - **Mestre** (VE) *Boffelli* C.so del Popolo, 32/c Tel.: 041/951247-5057812 Dr. Roberto Conforto - **Mestre Loc. Chirignago** (VE) *Computime* Via Miranese, 420 Tel.: 041/917566 - **Padova** *System Ros* P.za De Gasperi, 14 Tel.: 049/38412 Sig. Pasqualetto - **Portici** (NA) *Sisa* Via Canarde, 14 Tel.: 081/7755158 Ing. Di Maso - **Roma** *Data Office* Via Sicilia, 205 Tel.: 06/4742651 Dr. Triulzi - **Roma** *Expo* Via IV Novembre Tel.: 06/6783488 Sig. Ruffini - **Roma** *Personal Computer* P.za Pio XI, 26 Tel.: 06/6380353 Dr. Filippetti - **Roma** *Valde Adel* P.za Bainsizza, 3 Tel.: 06/316331-316676 Ing. Paolo Tropea - **S. Donà di Piave** (VE) *Dr. Spinazzè* P.za Rizzo, 63 Tel.: 0421/52548 - **San Gregorio di Catania** (CT) *Sistemi Sud Computers* Via Sgroppillo, 17 - Tel.: 095/493911 Sig. La Rosa - **Schio** (VI) *Bit* Via Roncoletto, 23 Tel.: 0445/28928 Sig. Bertoldi - **Schio** (VI) *Linea 4* Via del Cristo Tel.: 0445/28970 Sig. Zaffonato - **Tavernelle Altavilla** (VI) *Centro Informatica* Via Verona, 64 Tel.: 0444/573967-8 Sig. Todescan - **Treviso** *Informatica Tre* V.le della Repubblica, 19 Tel.: 0422/65993 Sig. Brugnara - **Trieste** *Ditta Murri* Via A. Diaz, 24/A Tel.: 040/306091 Sig. Migliarini - **Udine** *Michieli* V.le Ungheria, 64 Tel.: 0432/291835 Sig. Michieli - **Verona** *Computek Sistemi* V.le del Lavoro, 33 - Tel.: 045/509311 Sig. Farina - **Vicenza** *Alfa Data* Via Milano, 110 Tel.: 0444/31865-46481 Sig. Dal Dosso

Centro Direz. Colleoni
Palazzo Orione Ingr. 1
20041 Agrate Brianza (MI)
Tel.: 039/6363331
Telex: 326423 SIAV BC

Gli errori dovuti alla manipolazione di stringhe

Come interpretare correttamente alcuni messaggi di errore emessi dal vostro Commodore, e per non arrabbiarsi mai più.

SUCCESS

Molti sono gli errori che possono esser segnalati nella manipolazione delle stringhe. Li esamineremo, ovviamente, uno per uno ricordando, dapprima, ciò che si intende per "stringa".

Questa rappresenta un gruppo di caratteri qualsiasi, vale a dire alfanumerici e/o semigrafici e/o "speciali" (colore, reverse, spostamento cursore, tasti funzione f1-f8). Il numero dei caratteri che costituisce una qualsiasi stringa può variare da zero (cioè nessun carattere) fino a 255. Non appena si accende il computer qualsiasi stringa ha lunghezza nulla e non contiene, di conseguenza, alcun carattere. Esempio:

```
100 PRINT A$;A2$;XX$;X$(3)
```

Questo programma stampa... nulla proprio per il motivo appena detto. Si presti la massima attenzione al fatto che una stringa nulla non contiene alcun carattere. Abbiamo avuto modo di constatare che molti pensano (chissà perchè?) che una stringa nulla è costituita dal carattere di spazio, prodotto, in altre parole, dalla pressione della barra spaziatrice (detto, in gergo, "blank").

Proviamo ora il seguente micro-programma:

```
100 A$="": I=0
110 A$=A$+"."
120 PRINT I;A$
130 I=I +1: GOTO110
```

La sua funzione è quella di aggiungere, volta per volta, un carattere di punto (.) alla stringa denominata A$. All'inizio (riga 100) il valore di "I" e la variabile A$ vengono "annullati" (notare la mancanza di spazio tra i due caratteri di virgolette dopo A$=). Ogni volta che tale stringa diventa più lunga di un carattere (riga 110), viene stampato il numero di caratteri contenuti nella stringa e la stringa stessa, costituita da un numero sempre crescente di punti (riga 120). La riga 130 ha il compito di incrementare il valore

di "I" e di far continuare l'elaborazione alla riga 110. Non appena digitate Run (R) noterete ciò che segue:

```
RUN (R)
1 .
2 ..
3 ...
```

e così via, fino al valore di 255. Subito dopo verrà stampato il messaggio: "STRING TOO LONG ERROR IN 110". Se a questo punto, infatti, chiedete il valore di "I" (PRINT I) otterrete, come risposta, 256. Ciò vuol dire che una stringa non può contenere più di 255 caratteri (numerati da 1 a 255).

Esaminiamo ora un altro errore in cui è possibile incorrere:

```
100 INPUT A$
110 A=A$: GOTO 100
```

Questo programma non può girare perchè non è possibile (prima istruzione di riga 110) associare ad una variabile numerica una stringa alfanumerica anche se questa è formata soltanto da caratteri numerici. Esempio:

```
RUN (R)
? 185 (R)
TYPE MISMATCH ERROR IN 110
READY.
```

Ciò dimostra che nemmeno digitando soltanto caratteri numerici è possibile fare accettare istruzioni che associano variabili alfanumeriche a variabili numeriche. Per rendere operativo il programma è necessario trasformarlo nel modo seguente:

```
100 INPUT A$: X=VAL(A$)
110 A=X: PRINT A;A$: GOTO 100
```

Esempio:

RUN (R)
? 123 (R)
123 123
? 456QWER789 (R)
456 456QWER789
? QWERTY 123
0 QWERTY 123
(eccetera)

Da notare che con l'istruzione PRINT A (riga 110) ci aspetteremmo la visualizzazione del numero digitato a partire dal bordo sinistro dello schermo. Capita, invece, che il valore numerico viene stampato preceduto sempre da uno spazio bianco. Questo, in effetti, non è uno spazio superfluo ma rappresenta il posto che occuperebbe l'eventuale segno algebrico del valore digitato. Se, infatti, con lo stesso programma di prima digitate:

RUN (R)
? -123 (R)
-123

il valore, "corredato" di segno, viene stampato a partire dalla prima colonna dello schermo. Ciò è stato possibile grazie alla presenza del segno negativo che, come è noto, deve essere evidenziato. Quando un valore è invece positivo, è consentito sottintenderlo ed il computer non lo visualizza lasciando al suo posto, però, uno spazio bianco.

Con le stringhe tale procedura non viene applicata perchè non esistono stringhe positive o negative. Dal punto di vista della visualizzazione il computer, di conseguenza, tratterà le stringhe in modo diverso dalle variabili numeriche.

Digitate questi semplici esempi per rendervi conto di quanto asserito:

A=100:B=200:PRINTA;B (R)
Risultato:
 100 200

(Si noti la presenza dello spazio prima di 100 e quella di ben due spazi tra 100 e 200)

Provate ora:
A$="100": B$="200": PRINT A$;B$ (R)
Risultato:
100200

A parte l'assenza dello spazio d'inizio, si noti la totale assenza di blank tra le due stringhe (tanto che ne sembrano una sola) dovuta al fatto che un calcolatore non può sapere il reale significato del contenuto di una variabile stringa, qualunque essa sia.

Abbiamo fin qui visto in che modo trasformare una stringa in numero, grazie a VAL(X$), a patto, ovviamente, che la stringa contenga caratteri numerici.
Come avrete intuito è possibile effettuare l'operazione inversa, vale a dire la trasformazione di un valore numerico in stringa. L'istruzione che compie il miracolo è STR$. Vediamo subito un esempio:

A=100: A$=STR$ (A):PRINT A$ (R)
Risultato:
100

Noterete che ha subito una trasformazione anche il segno della variabile "A" dato che la visualizzazione della variabile A$ è spostata di un carattere a destra. Questo fatto è di fondamentale importanza nella gestione delle stringhe che "derivano" da valori numerici. Bisognerà infatti tener conto che, in questi casi, il primo carattere della stringa è sempre costituito da un carattere blank

(simboleggiante il segno positivo) oppure dal carattere meno (-) nel caso di conversioni di valori negativi.

Conclusioni sull'esperienza effettuata:

- *Non è possibile trattare stringhe lunghe più di 255 caratteri.*
- *Per associare ad una variabile numerica il valore di una stringa alfanumerica si utilizza l'istruzione VAL(X$).*
- *Per associare ad una variabile stringa un valore numerico "X" è indispensabile trasformarlo in stringa mediante STR$(X).*
- *Per esaminare le singole cifre che costituiscono un valore numerico risulta molto comoda la funzione STR$ a patto di ricordare che il primo carattere della stringa ottenuta rappresenta sempre il segno.*

La funzione stringa RIGHT$

Esaminiamo ora gli errori in cui è possibile incorrere utilizzando l'istruzione RIGHT$(X$,X).

E' bene ricordare che "Right", in inglese, significa "Destra" e, ovviamente, l'istruzione che contiene tale parola tenderà ad elaborare la parte destra di una qualsiasi stringa.

Ecco un esempio pratico:

```
100 A$="ABCDEFGHIL"
110 PRINT"QUANTI CARATTERI"
120 INPUT Q
130 PRINT RIGHT$(A$,Q)
140 GOTO 110
```

La riga 100 associa alla stringa A$ i primi dieci caratteri dell'alfabeto. Attenzione a rispettare la sintassi (soprattutto per ciò che riguarda le virgolette) in modo da evitare messaggi di Syntax Error.

La riga 110 e la 120 visualizzano la domanda (Quanti caratteri?) ed associano alla variabile "Q", grazie ad INPUT Q (riga 120) il valore digitato.

La riga 130 stampa, della parte destra (Right) della stringa interessata (A$) il numero di caratteri (Q) indicati con l'istruzione Input. Esempio:

```
RUN (R)
Quanti caratteri
? 4 (R)
GHIL
Quanti caratteri
? 8 (R)
CDEFGHIL
Quanti caratteri
? 0 (R)
(rigo vuoto)
Quanti caratteri
? 10 (R)
ABCDEFGHIL
Quanti caratteri
? 20 (R)
ABCDEFGHIL
Quanti caratteri
? 2880 (R)
ILLEGAL QUANTITY ERROR IN 130
```

Nel primo e nel secondo caso vengono stampati, rispettivamente, gli ultimi quattro e otto caratteri della stringa A$. Il terzo caso (Q=0) non stampa nulla. Interessante è il quinto caso in cui si chiede di visualizzare un numero di caratteri (20) maggiore di

quelli realmente presenti nella stringa A$. La risposta del computer non è la segnalazione di un errore, come si potrebbe pensare in un primo momento, ma la semplice stampa dell'intera stringa. Tale comportamento rimane fino a che il valore "Q" è compatibile con l'intervallo applicabile ad una stringa. Abbiamo già visto che il numero dei caratteri di una stringa può variare da zero (stringa vuota) a 255. Ciò spiega l'emissione del messaggio di "Quantità illegale" derivante dall'ultimo esempio riportato. Anche valori negativi generano lo stesso errore mentre valori decimali (come, ad esempio: 2.56) vengono troncati nella loro parte intera prima di eseguire l'istruzione.

Conclusioni sull'esperienza effettuata:

• *E' possibile individuare un numero "Q" di caratteri presenti alla destra di una qualsiasi stringa "X$" a patto che "Q" appartenga al "range" compreso tra zero e 255.*
• *Valori di "Q" incompatibili col range tipico delle stringhe provocano un "Illegal quantity error".*
• *Valori decimali di "Q" vengono egualmente accettati pur se limitati alla sola parte intera del valore assegnato.*

Un'esperienza perfettamente analoga, che il lettore può verificare da solo, riguarda l'utilizzo dell'istruzione LEFT$ che si riferisce alla parte sinistra (Left, in inglese) di una stringa. Modificate, a tale scopo, la sola riga 120 del programma precedente:

```
120 PRINT LEFT$(A$,Q)
```

Le conclusioni sull'uso di LEFT$ sono praticamente identiche a quelle tratte per RIGHT$ sostituendo, alla parola "Destra" il termine "Sinistra".

I messaggi personalizzati

Un modo di impedire la visualizzazione del messaggio e la conseguente, immediata interruzione del programma, consiste nell'inserire una sorta di "filtro", un controllo, insomma, capace di deviare l'elaborazione al momento opportuno. Utilizzeremo, a tale scopo, un'altra forma sintattica relativa al trattamento delle stringhe: LEN(X$) che consente di conoscere il numero dei caratteri presenti all'interno di una stringa. Il suo valore può variare, come è intuitivo, tra zero e 255.

Cancellate il programma precedente oppure modificatelo in modo da avere il seguente:

```
100 A$="ABCDEFGHIL"
110 L=LEN(A$)
120 B$="TROPPI CARATTERI"
130 C$="VALORE ILLECITO"
140 INPUT"N. CARATTERI";Q
150 IFQ>255 THEN PRINT C$:GOTO140
160 IFQ<0 THEN PRINT C$:GOTO140
165 IFQ>L THEN PRINT B$: GOTO 140
170 PRINT RIGHT$(A$,Q)
180 PRINT LEFT$(A$,Q): GOTO 140
```

La riga 100, come nel programma precedente, associa alla stringa A$ i primi dieci caratteri dell'alfabeto. Alla variabile "L" viene poi assegnato (riga 110) il numero di caratteri contenuti in A$. In questo caso particolare "L" sarà eguale a 10. Se il lettore inserirà un maggiore (o minore) numero di caratteri in A$, il valore di "L" varierà di conseguenza.

La riga 120 associa alla stringa B$ la frase-messaggio "Troppi caratteri"; analogamente la riga 130 associa a C$ "Valore illecito".

Nella riga 150 incontriamo un nuovo tipo di istruzione utilissima

e presente in moltissimi programmi. Ci riferiamo a: IF...THEN... (Traduzione: se...allora...).

Rinviamo il lettore al paragrafo che approfondisce lo studio di tale istruzione mentre qui daremo un cenno sommario sul suo funzionamento.

La riga 150 viene dal computer interpretata nel modo seguente:

- *Se (IF) si verifica il caso in cui il valore di "Q" è maggiore di 255 (Q>255), allora (THEN) devi eseguire tutte le istruzioni presenti dopo la parola THEN.*
- *Se, al contrario, non si verifica l'evento accennato (in altre parole: se "Q" è eguale oppure minore di 255), devi ignorare qualsiasi "cosa" sia presente dopo la parola THEN e continuare l'elaborazione passando alla riga Basic successiva (riga 160).*

Supponiamo che, al momento dell'elaborazione della riga 150, la variabile "Q", a causa di una nostra errata digitazione (in risposta all'Input di riga 140), contenga un valore maggiore di 255 (vedi dopo: primo esempio partico). Il computer, in tal caso, eseguirà tutto ciò che si trova dopo la parola THEN di riga 150. Col termine "tutto ciò" si intendono le istruzioni PRINTC$ e (dopo il doppio punto) il GOTO 140. La prima, visualizza la stringa B$ (vedi riga 120), mentre la seconda fa ritornare il programma alla riga 140.

Primo esempio pratico:

RUN (R)
Quanti caratteri? 400 (R)
Valore illecito
Quanti caratteri? -3 (R)
Valore illecito
(Eccetera)

Supponiamo ora che al momento dell'elaborazione della riga 150 la variabile "Q" contenga un valore minore di 255. In tal caso tutto ciò che è presente sulla riga 150 (dalla parola THEN fino alla fine della riga stessa), viene ignorato dal computer e l'elaborazione prosegue con la riga successiva che, nel nostro caso particolare, è numerata con 160. In questa è presente ancora un'altra istruzione del tipo IF...THEN. Stavolta, però, il computer eseguirà gli ordini presenti dopo THEN nel caso in cui si verifica la condizione imposta (Q<0) se, cioè, il valore di "Q" sia minore di zero, vale a dire negativo (preceduto dal segno meno).

Secondo esempio pratico:

Quanti caratteri? -5 (R)
Valore illecito
Quanti caratteri?
(Eccetera)

Se si digita, al momento dell'Input, (riga 140) un valore che non è maggiore di quello associato alla variabile "L" (determinato in riga 100) e che, nello stesso momento, non è minore di zero, il computer proseguirà con le righe successive (170-180) che visualizzano la parte destra e la parte sinistra della stringa A$. Subito dopo (Goto 140) riprende la solita domanda.

Terzo esempio pratico:

Quanti caratteri? 80 (R)
Troppi caratteri
Quanti caratteri? -3 (R)
Valore illecito
Quanti caratteri? 3 (R)
HIL
ABC
Quanti caratteri? 5 (R)
FGHIL
ABCDE
(eccetera)

Si noti che il terzo "filtro" di riga 165 è stato inserito non perchè in sua assenza si sarebbe verificato un errore, ma solo per illustrare un esempio di corretta programmazione.

Abbiamo infatti notato, in precedenza, che richiedere un numero di caratteri superiore a quello realmente disponibile nella stringa non genera un errore ma, tuttavia, potrebbe trarre in inganno l'utilizzatore. In altre parole se una stringa contenesse venti caratteri, e ne chiedessimo i primi 38, l'ordine verrebbe eseguito senza far notare l'incongruenza. Un apposito "filtro", che tenga conto di questa (lieve) carenza del Basic, consente all'utilizzatore di accorgersi di alcuni "difetti" di assegnazione di variabili.

Conclusioni sull'esperienza effettuata:

• *Considerando con attenzione la casistica relativa a tutte le manipolazioni di stringhe consentite, è possibile prevedere ed inserire messaggi di errore personalizzati non previsti dal Basic.*

• *Allo scopo di evitare interruzioni di programmi è necessario prevedere dei "filtri" software, vale a dire alcune righe di programma che impediscano opportunamente di incorrere in errori di calcolo.*

Le istruzioni Left$ e Right$ devono avere come "argomenti" (nome assegnato ai parametri posti tra parentesi) due variabili separate da una virgola. La prima deve essere una stringa, la seconda un valore numerico. Altri esempi di corretta sintassi sono i seguenti:

```
PRINT LEFT$("PAROLA",3) (R)
PAR
PRINT LEFT$("QUESTA"+"PAROLA",5*2) (R)
QUESTAPARO
```

```
PRINT LEFT$("ABC" +LEFT$("123",2),1+4)
ABC12
```

Come si può notare è possibile ricorrere a concatenazioni, funzioni di funzioni, calcoli complessi in un'unica istruzione. Dato che può sembrare complicato, forse è bene riportare alcuni esempi che, pur ricorrendo a più istruzioni, semplificano la comprensione del comando e, quel che più conta, facilita l'individuazione di eventuali errori commessi.

Comando "compatto"

```
A=1:B=4
PRINT LEFT$("ABC"+ LEFT$("123",2),A +B)
Risultato: ABC12
```

Comando suddiviso in più comandi:

```
A=1:B=4:C=A+B
A$="ABC":B$="123"
C$=LEFT$(B$,2)
D$=A$ +C$
PRINT LEFT$(D$,C)
Risultato: ABC12
```

La funzione MID$

L'ultima funzione che consente di esaminare una parte di una stringa alfanumerica, è MID$ che "estrae" una parte ben definita all'interno di una stringa.

Mentre RIGHT$ estrae la parte destra e LEFT$ quella sinistra, MID$ rintraccia un certo numero di caratteri presenti all'interno di una stringa.

Esempio:

```
100 A$="ABCDEFGHIL"
110 INPUT"DA QUALE CAR.";C
```

```
120 INPUT"N. CARATTERI";Q
130 PRINT MID$(A$,C,Q):GOTO 110
```

Questo listato, molto simile a quello visto in precedenza, necessita di due parametri richiesti mediante altrettante istruzioni del tipo Input. La prima, che richiede un valore numerico da associare alla variabile "C", chiede quale è il carattere di A$ che dovrà in seguito considerare come primo. Il secondo Input (variabile "Q", riga 120) terrà conto del numero di caratteri da considerare.

Esempio:

```
RUN (R)
Da quale carattere? 3 (R)
N. Caratteri? 4 (R)
CDEF
Da quale carattere? 4 (R)
N. Caratteri? 0 (R)
(rigo vuoto)
Da quale carattere? 120 (R)
N. caratteri? 120 (R)
(rigo vuoto)
Da quale carattere? 0 (R)
N. Caratteri? 2 (R)
ILLEGAL QUANTITY ERROR
```

Anche con MID$ valori privi di significato sono egualmente accettati dal computer. Generano invece la segnalazione di errore illegale (e conseguente interruzione del programma) sia i valori esterni al range delle stringhe (0-255) sia l'assegnazione di valori minori di uno al secondo argomento contenuto tra parentesi.

Anche in questo caso il lettore potrà inserire un filtro seguendo le stesse modalità applicate per il programma precedente. E' ovvio che, in questo caso, saranno necessari più controlli e, magari, un maggior numero di messaggi personalizzati allo scopo di meglio interpretare incongruenze di elaborazioni.

L'istruzione MID$ accetta anche due soli argomenti. Quello che può essere omesso è il terzo,relativo al numero di caratteri da considerare. In questo caso la "sottostringa" (cioè la parte di una stringa) sarà costituita da tutti i caratteri a partire da quello indicato dal secondo argomento. Cancellate il programma presente in memoria oppure modificatelo come segue:

```
100A$="ABCDEFGHIL"
110 INPUT"DA QUALE CAR.";C
120 PRINT MID$(A$,C):GOTO 110
```

Esempio:
RUN(R)
Da quale carattere? 6 (R)
FGHIL
Da quale carattere? 10 (R)
L
Da quale carattere? 0 (R)
Illegal Quantity Error in 110

Omettiamo, stavolta, sia di spiegare il motivo della segnalazione di errore, sia il modo di inserire filtri prima di eseguire il comando.

Conclusione sull'esperienza effettuata:

- *Prima di rendere operativo un comando che gestisce le stringhe è opportuno esaminare, a parte, gli argomenti del comando stesso.*
- *Controllare attentamente che gli argomenti siano tutti presenti: due, nel caso di* ***Right$*** *e* ***Left$****; due, oppure tre, nel caso di* ***Mid$****.*
- *Se viene segnalato un Illegal Quantity Error, ricercate l'errore nei parametri posti tra parentesi. Assicuratevi, in altre parole, che gli argomenti si riferiscano a stringhe non nulle oppure a valori che non rispettano i limiti imposti dal range (0-255).*
- *Nel caso di* ***Mid$*** *il secondo argomento può variare tra uno e 255 (e non tra 0 e 255).*

Le funzioni ASC(X$) e CHR$(X)

Un'altra istruzione che, se mal utilizzata, genera un messaggio di errore, è rappresentata da ASC(X$) che "restituisce" il valore numerico (del codice ASCII) relativo al primo carattere della stringa X$.

Allo scopo di chiarire le idee è bene ricordare che esiste una tabella, accettata da quasi tutti i fabbricanti di computer, che associa ad ogni carattere alfanumerico, rappresentabile con un calcolatore, un numero intero compreso tra 0 (zero) e 255. La tabella in questione (Standard ASCII) è riportata in un'appendice di un qualsiasi manuale di istruzioni Commodore. Ad esempio, al carattere alfabetico "A" è associato il valore 65, al carattere "9" il valore 57 e così via. Non tutti i valori sono però comuni a tutti i computer prodotti. Solo quelli compresi tra 32 (carattere blank) fino a 95 (freccia a sinistra) oltre a qualche altro (come il tasto Return, che vale 13) sono accettati da tutti i produttori di calcolatori. Certi valori, per alcuni computer, non rappresentano nulla mentre per i Commodore possono avere un preciso significato, e viceversa.

Prima di esaminare la funzione ASC, diamo uno sguardo alla funzione legata "direttamente" al codice ASCII: la funzione CHR$.

Questa deve essere associata ad un argomento numerico (0-255) ma non tutti forniscono un risultato concreto.

Provate infatti a digitare questo semplice minilistato:

```
100 INPUT"VAL.(0-255)";V
110 PRINT CHR$(V)" PAROLA"
120 GOTO 100
```

Non fa altro che assegnare il valore "V" all'argomento di CHR$. Il messaggio " Parola" posto subito dopo (notare il carattere blank prima di "P"), serve unicamente ad osservare eventuali "effetti" altrimenti invisibili.

Se, dopo aver digitato Run (R), digitate, alla domanda "Val.?" un valore compreso tra 0 e 255, otterrete, a seconda dei casi, risposte differenti più o meno comprensibili.

Valore Effetto

65 A Parola
66 B Parola
68 C Parola
1 Parola
13 (rigo vuoto)
Parola
144 Parola
(col.nero)
5 Parola
(col.bianco)
147 (canc.schermo)
Parola
18 Parola
(scritto in
reverse)
eccetera.

Ciò significa che al valore 65 corrisponde "A", a 144 corrisponde il colore nero, a 18 il reverse eccetera. Alcuni valori, invece, non hanno alcun significato (come 1, 2, 3 ed altri).

A parte l'errore segnalato dal computer nel caso si superi il range (0-255: Illegal Quantity), c'è da segnalare un errore "logico" dovuto al fatto che è possibile inviare un carattere speciale di colore eguale a quello del fondo dello schermo. Se in un particolare momento avete il video colorato in bianco e digitate:
PRINT CHR$(5)"PAROLA"

Il comando verrà eseguito ma il messaggio "Parola" (e tutti gli altri che eventualmente lo seguono) saranno colorati in bianco col

risultato di essere invisibili. Ciò vale anche con caratteri blu su fondo blu, verdi su fondo verde eccetera.

Per ciò che riguarda la funzione ASC c'è da dire che si può applicare solo a stringhe costituite da almeno un carattere. Nel caso di stringhe nulle il messaggio di errore è il consueto "Illegal Quantity". Esempio:

```
A$="":PRINT ASC(A$) (R)
Illegal Quantity Error
```

Nel caso, invece, di stringhe costituite da più caratteri, la funzione ASC si riferirà automaticamente al primo di questi. Esempio:

```
A$="PIPPO":PRINT ASC(A$) (R)
Risposta: 80
```

80 è infatti il codice ASCII del carattere "P".

Alcuni casi strani.

Possono capitare strane circostanze, apparentemente inspiegabili. Ad un certo punto di un'elaborazione, chiedendo il codice di una stringa che riteniamo di conoscere in modo inequivocabile, possono verificarsi risposte insensate.

Supponiamo, ad esempio, di digitare: Print A$ (R) e di ottenere in risposta: PROVA. Ci aspetteremmo, con Print ASC(A$) il valore 80 (codice di "P" di "Prova"). Può capitare di avere altre riposte, come ad esempio 13 oppure 17 o altre ancora.

Ciò è dovuto al fatto che ciò che noi crediamo sia una stringa formata solo da caratteri alfanumerici, è in realtà il risultato di una concatenazione ben più complessa. Per meglio comprendere qunato asserito, digitate ciò che segue:

```
A$=CHR$(13):B$=A$+"PIPPO" (R)
PRINT B$:PRINT ASC(B$) (R)
```

Risultato:
(rigo vuoto)
PIPPO
 13

Con un po' di pratica sarà facile risalire al reale contenuto di una stringa. Nei casi più difficili suggeriamo una tecnica semplice da ricordare. Supponiamo che sia A$ la stringa che presenta qualche problema. Digitate, in questi casi, quanto segue:

```
L=LEN(A$):
FOR I=1 TO L:
PRINT ASC(MID$(A$,I,1));:
NEXT
```

Verranno visualizzati i codici ASCII dei singoli caratteri costituenti la stringa.

Un altro "trucco" per esaminare con calma, oltre ai caratteri alfanumerici, anche i caratteri "speciali" dei colori e del cursore eventualmente presenti, è quello di far stampare il carattere di virgolette, CHR$(34), prima della stringa interessata. Esempio:

```
A$=CHR$(147)+CHR$(17):
B$=A$+"PROVA":
PRINT CHR$(34);B$
```

Risultato: visualizzazione del carattere virgolette (") seguito dal cuore in campo inverso (simbolo della cancellazione dello schermo), dalla "Q" in campo inverso (cursore in basso) e, finalmente, dalla parola "PROVA".

Il semplice comando PRINT B$ non avrebbe consentito un esame così approfondito.

Conclusioni sul trattamento delle stringhe:

• *LEN(X$) restituisce il valore intero compreso tra 0 e 255, rappresentante il numero di caratteri che costituiscono la stringa X$. Se il*

valore che risulta sembra maggiore di quello che ci aspettiamo, significa che alcuni caratteri della stringa sono speciali oppure, pur presenti, non hanno alcun significato. Nel caso venga visualizzato un Syntax Error la causa è da ricercarsi nell'argomento che, sicuramente, non è una variabile stringa ma una variabile numerica. Esempio: LEN(X).

- *STR$(X) trasforma in stringa il valore della variabile "X". STR$(X$) genera un Syntax Error.*
- *VAL(X$) restituisce il valore numerico della stringa X$ relativo ai primi caratteri di X$ che siano numerici. Se il primo carattere di X$ non è numerico, il valore che ne risulta sarà sempre nullo.*
- *CHR$(X) restituisce il simbolo ASCII corrispondente al valore "X" (compreso tra 0 e 255). CHR$(X$) genera un Syntax Error: l'argomento deve essere un valore numerico e non una stringa.*
- *ASC(X$) restituisce il valore ASCII del primo carattere della stringa X$. ASC(X) genera un Syntax Error per i motivi prima visti.*

Errori nei cicli IF...THEN

Abbiamo già avuto modo di incontrare un ciclo If...Then. Vedremo ora di approfondire lo studio di queste importantissime istruzioni.

Gli errori che è possibile commettere, oltre quelli banali di trascrizione e, quindi, di sintassi (TEN invece di THEN e simili) sono soprattutto gli errori "logici", quel tipo di errori, cioè, che non possono in alcun modo essere individuati dal computer e che solo un attento esame del programma può evitare.

Iniziamo, come al solito, dai casi più semplici, compresi gli errori di sintassi che è possibile commettere.

Ci "ispireremo", a tale scopo, al programma già utilizzato in precedenza. Il problema da risolvere è il seguente: introdurre (mediante Input) un numero nel calcolatore e fare apparire un

messaggio che ci indichi se tale valore, rispetto a zero è minore (cioè negativo), eguale, oppure maggiore (cioè positivo). E' un problema decisamente stupido ma, grazie alla sua brevità, consentirà tuttavia di esaminare i vari tipi di errore che potremmo commettere in programmi più complessi.

```
100 A$="E' MINORE DI ZERO"
110 B$="E' UGUALE A ZERO"
120 C$="E' MAGGIORE DI ZERO"
130 INPUT"DIGITA NUMERO";A
140 IF A<0 THEN JJJFJ
150 IF A=0 THEN PRINT A;B$:GOTO130
160 IF A>0 THEN PRINT A;C$:GOTO130
```

Esaminiamolo riga per riga: nelle prime tre (100, 110, 120) alle variabili stringa A$, B$ e c$ sono associati tre messaggi da visualizzare in seguito al momento opportuno. La riga 130 è il noto Input che, dopo aver fatto apparire il messaggio "Quale numero?" associerà alla variabile "A" il valore digitato.

La riga 140 contiene un errore di sintassi (JJJFJ), inserito intenzionalmente, la cui funzione esamineremo in seguito. La parte presente dopo il Then di riga 150 "funzionerà" a patto che (If A=0) il valore di "A" sia nullo.

La stessa cosa avverrà alla seconda parte della riga 160 nel caso in cui "A" sia maggiore di zero. Vediamo ora come gira il programma proposto:

```
RUN (R)
Digita numero? 85 (R)
85 è maggiore di zero
Digita numero? 0 (R)
0 è eguale a zero
Digita numero? -45 (R)
Syntax error in 140
READY.
```

Il messaggio di errore, comparso digitando il valore negativo -45, è dovuto al fatto che in questo caso, verificata la condizione della riga 140 (A<0), il computer si appresta ad elaborare le istruzioni presenti dopo la parola Then. Poichè ciò che è presente dopo Then non ha alcun significato sintattico (JJJFJ), viene generato un Syntax Error.

Ciò dimostra che il Basic è un linguaggio "Interprete". In un programma possono quindi esistere errori di diverso genere, oltre che di sintassi, che vengono evidenziati solo se le linee che li contengono sono interessate nello svolgimento del programma. Alcuni errori, addirittura, (come avremo modo di vedere) possono non essere individuati in nessun caso!
Modifichiamo ora la riga 140 come si conviene:

```
140 IF A<0 THEN PRINT A;A$:GOTO130
```

Il programma girerà perfettamente. Vedremo ora di esaminare un metodo spesso adoperato per "risparmiare" istruzioni che però, se usato in modo superficiale, può generare errori "logici".

Abbiamo detto che, se si verifica la condizione imposta da IF, verranno eseguiti tutti gli ordini presenti dopo "Then". Dopo aver eseguito l'ultimo di essi, il programma proseguirà interpretando (cioè eseguendo) gli ordini presenti nella riga Basic immediatamente successiva a quella che contiene If...Then.

Spieghiamoci meglio: utilizzando lo stesso programma di prima supponiamo di digitare un qualsiasi valore negativo (come: -5). Poichè, appunto, "A" è minore di zero, sarà "attivata" la linea 140 e il computer passerà ad esaminare ed eseguire tutti gli ordini presenti dopo Then. Esaminiamoli uno per uno.

Il primo di essi è Print e, di conseguenza, il calcolatore si appresta a visualizzare ciò che troverà dopo Print, vale a dire il contenuto della variabile "A" e, subito dopo (grazie al punto e virgola) il contenuto della variabile A$ che rappresenta il messaggio di riga 100. Dopo A$ è presente il carattere di doppio punto (:) che "interrompe" l'azione di Print e rende operativo il comando

successivo. Questo è un Goto 130 che impone al programma di proseguire con la linea Basic indicata. In questo caso, e lo notiamo esplicitamente, il programma prosegue inesorabilmente con la linea 130 e NON con la linea successiva (150). Modifichiamo ora il programma e vediamo in che modo è possibile, rispettando il filo "logico", risparmiare istruzioni:

```
100 A$="E' MINORE DI ZERO"
110 B$="E' UGUALE A ZERO"
120 C$="E' MAGGIORE DI ZERO"
130 INPUT"DIGITA NUMERO";A
140 IF A<0 THEN PRINT A;A$
150 IF A=0 THEN PRINT A;B$
160 IF A>0 THEN PRINT A;C$
170 GOTO 130
```

Se digitiamo Run e (R), notiamo che l'effetto è identico al precedente ma il "ragionamento" seguito dal computer è lievemente diverso pur se conduce agli stessi risultati. Il perchè è presto detto.

Supponiamo di digitare, al momento dell'Input di riga 130, un valore positivo come 32. Non appena premiamo il tasto Return, il programma continua con la riga 140 che impone (If A<0) un confronto tra il valore digitato e lo zero. Poichè 32 è maggiore di zero, come abbiamo visto, non viene presa in esame la parte successiva a Then. L'elaborazione, quindi, prosegue con la linea successiva a 140 (riga 150). Anche in questo caso il confronto fornisce esito negativo e l'elaborazione prosegue con l'esame del contenuto di riga 160. Questa, finalmente, si "accorge" che il confronto ha esito positivo ed esegue senza indugio il comando Print A seguito dal punto e virgola. Verrà quindi stampato il numero 32 e, subito dopo (grazie al punto e virgola) il contenuto di A$. Poichè il rigo Basic non contiene alcuna altra istruzione, il computer prosegue con la riga Basic successiva. Questo fatto deve esser tenuto ben presente perchè molto spesso non si individuano

le cause di malfunzionamenti di programmi proprio per la leggerezza con cui vengono utilizzate istruzioni If...Then. Vediamo ora un esempio.

Modifichiamo il programma di prima introducendo una piccola modifica logica dovuta al seguente ragionamento: se un numero non è negativo nè nullo, ne deduciamo che DEVE esser positivo. Il primo programma di questo paragrafo può dunque esser modificato come segue:

```
100 A$="E' MINORE DI ZERO"
110 B$="E' UGUALE A ZERO"
120 C$="E' MAGGIORE DI ZERO"
130 INPUT"DIGITA NUMERO";A
140 IF A<0 THEN PRINT A;A$:GOTO130
150 IF A=0 THEN PRINT A;B$:GOTO130
160 PRINT A;C$:GOTO130
```

Se, dunque, digitiamo un valore negativo o nullo il programma segue lo stesso percorso già noto. Se, però, digitiamo un valore positivo, verranno effettuati (senza esito) i confronti delle righe 140 e 150 e il programma proseguirà alla linea 160 raggiungendo gli stessi scopi visti prima nonostante sia assente un confronto del tipo IF A>0. Tale confronto, lo ripetiamo, è implicito e si fonda sul ragionamento che se "A" non è minore nè eguale deve esser necessariamente maggiore di zero.

Se però ricorriamo anche all'eliminazione dei Goto 130 (presenti nelle righe 140 e 150), come avevamo suggerito nel secondo esempio, succederà che... provate voi stessi e tenete bene a mente le conseguenze che possono derivare da uno scorretto uso di If...Then.

Altre forme di If...Then

Poichè, per i motivi visti prima, in caso di confronto positivo vengono eseguite tutte le istruzioni presenti dopo la parola Then,

se ne deduce che il numero di comandi, istruzioni, funzioni e confronti sono costretti a "coabitare" sulla stessa riga Basic.

In alcuni casi, purtroppo, gli ottanta caratteri della riga Basic non bastano per contenere "tutto ciò" che deve esser elaborato in caso di esito positivo. In questi casi si può ricorrere a diversi "trucchi". Supponiamo che un nostro programma, dopo aver visualizzato un messaggio ("Valori compresi...) debba proseguire alla linea 3800 soltanto se si verificano entrambi i seguenti due casi:

1/ se la variabile "XR" è compresa tra -0.5624 e 4567.34
2/ se la variabile "H" è maggiore di 456 ma minore di 567.89.

Una linea che tenga conto di queste due condizioni dovrebbe essere la seguente:

```
1200 IF (XR>=-.1234 AND XR<=5678.99) AND (H>456
AND H>567.89) THEN PRINT "VALORI COMPRESI
NELL'INTERVALLO LECITO": GOTO3800
```

Purtroppo, come si può notare, a parte la scarsa "leggibilità", queste istruzioni non possono esser contenute in una semplice riga Basic. E' quindi necessario "spezzarla" in modo da esser più comprensibile e, contemporaneamente, più "logica". Supponendo di non aver utilizzato nelle righe precedenti le variabili "Q", "W", "E", "R", Y$ ecco un modo per risolvere il problema:

```
1180 Q=-0.1234: W=5678.99: E=456: R=567.89
1190 Y$="VALORI COMPRESI NELL'INTERVALLO
     LECITO"
1200 IF (XR>=Q AND XR<=W) AND (H>E AND H<R)
THEN PRINT Y$: GOTO 3800
```

Possono capitare, comunque, tante e tali condizioni che nemmeno il ricorso alla tecnica descritta può superare l'ostacolo. In

questi casi si deve ricorrere alle subroutine che, come è noto, possono esser lunghe quanto si vuole. Lo stesso esempio di prima può esser risolto, ricorrendo alla tecnica delle subroutine, nel modo seguente:

```
1180 Q=-0.1234: W=5678.99: E=456: R=567.89
1190 Y$="VALORI COMPRESI NELL'INTERVALLO
     LECITO"
1200 IF XR>=Q AND XR<=W THEN GOSUB 50000:
     IF UX THEN 3800
................
................
50000 UX=0: IF H>E AND H<R THEN PRINT Y$: UX=1
50010 RETURN
```

Questa tecnica, che consigliamo ai lettori di studiare in dettaglio, consente di effettuare infiniti controlli e di spezzare in più righe gruppi di istruzioni che dovrebbero coesistere in una sola riga.

Ci limiteremo ad evidenziare solo alcune peculiarità della tecnica utilizzata.

Solo se XR contiene un valore compreso nell'intervallo "Q-W" viene effettuato un salto alla subroutine 50000. La seconda parte del confronto ("H" compreso tra "E" ed "R") viene effettuata proprio in questa riga. Si noti che la variabile "UX" assume la funzione di deviatore: Se, infatti, la variabile "H" non è compresa nell'intrevallo imposto, si "ritorna" alla riga 1200 e, dato che "UX" vale zero, la seconda parte di quest'ultima riga viene ignorata (come deve essere).

Se, invece, il confronto tra "H" e l'intervallo stabilito conduce ad esito positivo, non solo viene stampato il messaggoo Y$ ("Valori compresi...") ma il deviatore "UX" viene posto eguale ad uno. Al ritorno dalla subroutine, stavolta, il contenuto di "UX" farà proseguire il programma alla riga 3800.

Si noti che l'espressione IF UX THEN è perfettamente lecita benchè sembri priva di un confronto "ufficiale". In questo caso, infatti, la forma sintattica abbreviata significa: "Se UX (esiste) allora...". Il termine "esiste" sta per "è diverso da zero (positivo o negativo che sia)". Anche la forma sintattica abbreviata (Then 3800) non è causa di Syntax Error in quanto, dopo Then, è consentito sottintendere il comando "Goto".

L'errore "Division by zero"

Su questo errore ci soffermeremo pochissimo ricordando che non è possibile, con la matematica "normale" risolvere una divisione del tipo: 589/0 in cui, cioè, il denominatore sia nullo. Si può comunque dire, ad "intuito", che il risultato di un'operazione con divisore nullo sia infinito.

Il computer, se riscontra casi come quello accennato, emette un "Division by zero error" e interrompe l'elaborazione.
Se, ad esempio, il calcolatore emette il messaggio in questione con un:
"Division by zero error in 1540"
digitate List 1540 (R). Troverete sicuramente un'espressione del genere:
1540 ...A/(R+Y)...

Se ne deduce che il denominatore della frazione (RtY) è nullo. Ciò può capitare in due casi: sia "R" che "Y" sono nulli; "R" è uguale a "Y" ma hanno segni opposti tra loro.

Termina con queste note la nostra rassegna sull'esame dei più importanti messaggi di errore che è possibile osservare in esperimenti di programmazione.

Sarebbe opportuno che il lettore, continuando per proprio conto, sviluppi programmi sempre più complessi approfittando delle segnalazioni di errore per maturare esperienze sempre più valide.

Avviso ai lettori

Avete scoperto qualche incongruenza dei computer Commodore oppure qualche tecnica particolare di programmazione in Basic?

Scrivete o telefonate (02/8467348) alla Redazione di CCC proponendo il vostro lavoro.

Avete un problema di (semplice) programmazione che non riuscite a risolvere a causa di inspiegabili segnalazioni di errore?

Utilizzate la scheda posta in fondo alla rivista: se il problema è di interesse generale sarà data risposta attraverso le pagine della "Posta".